RC 20480 (06/17/96)
Computer Science

IBM Research Report

Video Conferencing over Packet-Based Networks

M. Willebeek-LeMair and Z.-Y. Shae

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, New York
mwlm, zshae @watson.ibm.com

LIMITED DISTRIBUTION NOTICE

This report has been submitted for publication outside of IBM and will probably be copyrighted if accepted for publication. It has been issued as a Research Report for early dissemination of its contents. In view of the transfer of copyright to the outside publisher, its distribution outside of IBM prior to publication should be limited to peer communications and specific requests. After outside publication, requests should be filled only by reprints or legally obtained copies of the article (e.g., payment of royalties).

Research Division
Almaden · T.J. Watson · Tokyo · Zurich

Video Conferencing over Packet-Based Networks

Marc H. Willebeek-LeMair and Zon-Yin Shae

IBM Research Division
T.J Watson Research Center
Yorktown Heights, NY 10598, USA
mwlm, zshae@watson.ibm.com

Abstract

The Internet explosion is driving the need for new collaboration tools which will enable two or more users to share data, audio and video. The real-time packet-based solutions which are emerging differ considerably from the circuit switch solutions which have existed for some time now. In this paper we present one such packet-based approach, the Multimedia Multiparty Teleconferencing (MMT) system, which was fully implemented as a research prototype. Using MMT as an example, we address some of the fundamental issues related to videoconferencing systems in a packet-based environment and discuss the differences with the traditional circuit switch approaches, namely the ITU H.320 standard. In particular, MMT is a distributed solution while H.320 is centralized. The use of multicast and a novel video mixing technique to facilitate the distributed solution are presented. Furthermore, MMT audio and video streams are susceptible to congestion and packet loss in the shared media packet-based environment while H.320 streams use dedicated connections. As such, synchronization and rate control schemes for the packet-based system are presented.

Keywords: videoconferencing, packet video, distributed MCU, video mixing, rate control

1. Introduction

The recent overwhelming interest and awareness of the Internet is increasing the need for, and consequent development of, collaborative computing tools of which videoconferencing is an important component. Videoconferencing solutions and standards have existed for some time now in the circuit-switched N-ISDN environment. Many new solutions are emerging which are designed for packet-based Intranet and Internet environments and a series of standards are being developed.

Two distinct classes of videoconferencing systems exist today. In one class are the circuit-switched (e.g., Narrowband ISDN or the Switched-56Kbps phone line) types of solutions. In the other class are the packet-based network (e.g., Ethernet and Token Ring, or more generally, IP network) solutions, which endeavor to carry real-time traffic over existing computer communications networks. Solutions for the circuit switched and packet-based networks differ considerably. These network technologies differ in terms of bandwidth usage and availability, connection vs connectionless communication paradigms, jitter, latency, packet-loss (error) characteristics, etc. As a consequence, the solutions for videoconferencing are adapted to exploit the capabilities/characteristics of each network. These differences include the encoder/decoder (CODEC) technology for video compression and decompression, the methods used to guarantee network performance, and in the provisions within the end-stations to handle the real-time traffic.

The first videoconferencing standard to emerge was the International Telecommunications Union H.320 [1]. This standard was defined for a circuit switched Narrowband ISDN environment at bandwidths ranging from 64 kbps up to over 2 Mbps (often termed the px64 standard in reference to the type of ISDN connection). The H.320 Standard consists of several components: H.261 [2], the video CODEC for audiovisual services at px64 kbits. G.711, G.722, G.728 (optional) the audio CODECs for audiovisual

services at px64 kbps. H.221 [3] the frame structure for a 64 to 2 Mbps channel in audiovisual teleservices. H.242 [4], the system for establishing communication between audiovisual terminals using digital channels up to 2 Mbps. H.230 [5] the frame-synchronous control and indication signals for audiovisual systems. H.231 [6], the multipoint control units for audiovisual systems using digital channels up to 2 Mbps. T.120 [7], the data protocols for multimedia conferencing. A large number of commercial desktop conferencing products based on the H.320 standard are available today and the vast majority of installed room-based systems conform to this standard (without T.120).

The ITU Standards have now begun addressing videoconferencing in different network environments, such as the H.324 for POTS, the H.323 for LANs, and H.310 suite for ATM. H.310 is a multimedia terminal recommendation for use over ATM networks and uses equivalent ISO MPEG-2 standards for video (ITU-T H.262) and systems (ITU-T H.222.0). H.323 is the recommendation for H.320 over LANs. H.324 is the standard for very low bit-rate telephony videoconferencing which uses an improved video compression (H.263) and audio compression (G.723) standard.

In all of these emerging standards, interoperability with the H.320 standard is mandatory. This interoperability, however, may be achieved in many ways. The various alternatives differ in the degree to which they embrace the H.320 model. Various aspects of the H.320 standard are clearly reusable in these other environments while others are not. For example, the audio and video compression algorithms or variations of them are being adopted by most of the evolving standards, however, different multiplexing schemes are being developed to better suit each network.

One aspect of the H.320 standard which is less clear as to its applicability to these other network environments is the centralized approach to multiparty conferencing (Figure 1 (a)). The H.320 standard defines a central conference server called a Multipoint Control Unit (MCU) to enable multiparty calls. Each participant in the call connects directly to the MCU, which then controls the conference. This paradigm is clearly suited to the point-to-point connectivity nature of ISDN networks. However, it is not the obvious choice for shared LANs or packet switched networks which support multicast capabilities. This issue is being debated by the H.323 standard committee.

In this paper we present a packet-based videoconferencing solution called Multimedia Multiparty Teleconferencing (MMT), which has been prototyped at the IBM Watson Research [8-15]. The goal of the MMT project was to design and implement a fully distributed multiparty videoconferencing system which leverages the multicast feature and higher bandwidth available in a packet-based network environment. As implemented, the MMT system allows each user to individually select which participants to view or hear. Multiple audio streams are audible simultaneously and multiple video streams may be viewed simultaneously.

Unlike the circuit-switched environment which employs a centralized MCU to enable the multiparty conferencing capability, the MMT system achieves this in a distributed manner (see Figure 1(b)). This distributed packet-based solution has several key features which distinguish it from the circuit-switched network centralized approach.

1. Multicast is used to distribute data from each station to other stations in the conference.
2. The audio and video mixing functions are performed within each individual end-station. To be practical, this demands a low cost solution. The MMT system composes the video from multiple sources in the compressed domain. This is a patented technique [16] and was originally described in [14].
3. Synchronization of sender and receiver CODECs in the absence of a network reference clock with robustness to packet jitter and loss.
4. A dynamic rate control scheme which allows the video encoder to adapt its transmission rate to different network conditions.

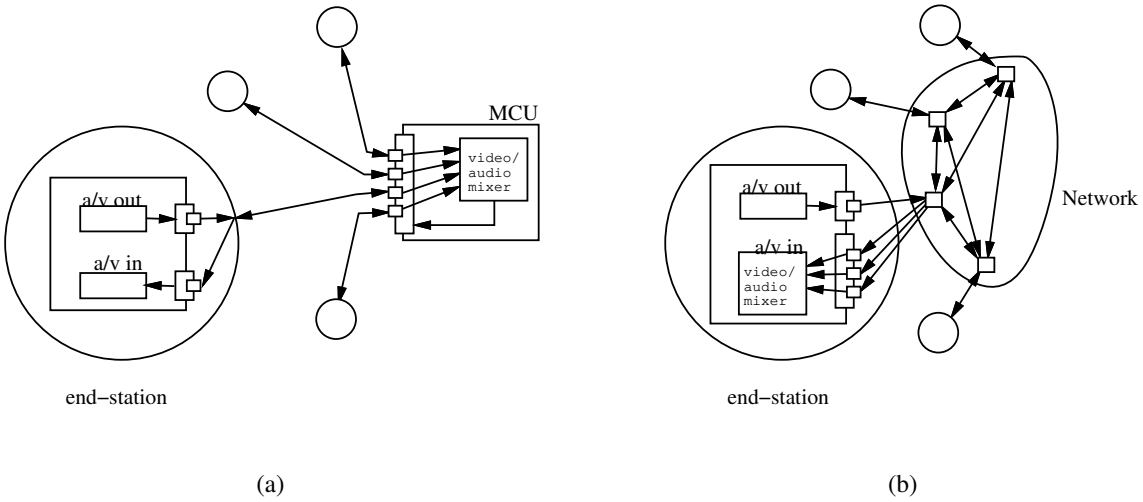

Figure 1. *(a) A centralized circuit switched videoconferencing model and (b) a distributed packet-based videoconferencing model.*

To illustrate the differences between a circuit switched videoconferencing solution and a packet-based solution, we use the ITU H.320 standard to represent the circuit switched class of solutions and MMT to represent the packet-based class of solutions. In Section 2 we describe the circuit switched solution with particular emphasis on the use of MCUs for multiparty conferencing. In Section 3 we discuss the primary issues which differentiate a packet-based network environment from a circuit switched one and present the MMT solution in Section 4. Section 5 is dedicated to the issue of rate control and includes a summary of experimental results obtained using the MMT system. A conclusion and brief summary of open issues is presented in Section 6.

2. The Circuit Switch Solution

2.1 The Circuit Switched Network Environment

In a circuit switched network all connections flow over dedicated point-to-point links. These connections are of constant bandwidth negotiated as part of the call setup phase. Call admission is conducted by either accepting or rejecting a call upon initialization. Once established, a connection is dedicated between the end-points which have established the connection and the bandwidth is guaranteed for the duration of the call. Hence, so long as the amount of information (audio, video, and data) transmitted by the source is less than or equal to the connection bandwidth, it will flow unobstructed to the destination.

To insure that the transmission rate at the sending station conforms to the connection bandwidth, the source synchronizes its CODEC to the network clock. Similarly, the receiving station synchronizes to he network clock to accurately render the incoming audio and video at the same rate as it is generated by the sender. This is useful in preventing buffer overflows and underflows and allows systems to be designed with minimum network latency. One drawback of this approach is that the video bit rate is forced to be constant to efficiently utilize the connection bandwidth. This introduces additional delay in the compression engine since a rate control buffer must be maintained to provide feedback to the encoder (see Figure 2). Also, video compression is highly content dependent and therefore, in order to maintain a constant bit rate, variations in the video quality (either in terms of frame rate or quantization of the video pixel values) is necessary.

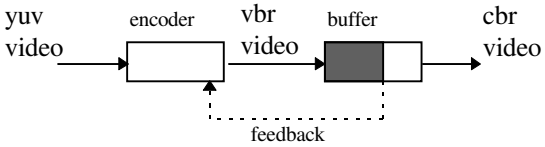

Figure 2. *Compression engine for a constant bit-rate H.261 encoder in a circuit switch environment. Raw yuv video is input to the encoder which produces a somewhat variable bit-rate (vbr) output which is placed in a buffer from which video is extracted at a constant bit rate (cbr). Feedback of the buffer fill-level is used by the encoder to adjust its compression parameters.*

2.2 The Centralized MCU Approach for Multiparty Conferencing

In the circuit switched environment a centralized approach is used to achieve a multiparty conference. In this approach, all conference streams are transmitted to a central server which then distributes composed and/or selected streams to the multiple participants. In the ITU H.320 terminology, this central server is known as a Multipoint Control Unit (MCU). The functions performed by an MCU are many and range in complexity. Basic MCU designs do little more than select a video stream from multiple incoming streams to be distributed to all conference participants. More complex designs perform a composition of multiple video streams before distributing the video stream to various participants. To illustrate the differences between various MCU capabilities along with the tradeoffs between the complexity of these functions, the four MCU configurations shown in Figure 3 are discussed.

2.2.1 Select and Broadcast

The MCU configuration in Figure 3 (a) is a simple design which selects one of multiple incoming streams to be broadcast to all participants. The criteria used to select the video stream to be distributed may vary. In a user controlled system, a designated user (conference manager) may send control information to the MCU to select the video source. Automatic systems may select the video source which corresponds to the active voice signal. The sensitivity of the selector must be adjusted to prevent an unintelligible thrashing between different sources. Synchronization is required when switching between sources to insure that the switching occurs at frame boundaries and, in some cases, that a constant bit rate is maintained.

2.2.2 Destination Select

The MCU configuration in Figure 3 (b) is slightly more sophisticated in that each recipient may independently select which video stream to receive. This requires a switch within the MCU with multicast capability. Signaling between each destination and the MCU is required to implement the selection. As with the Select and Broadcast configuration, switching between video streams must be synchronized to frame boundaries.

2.2.3 Compressed Composition

The MCU configuration in Figure 3 (c) performs a composition of the incoming video streams in the compressed domain. Techniques for performing this composition of compressed video are presented in [14]. The resulting video stream bandwidth may be a multiple of the incoming bandwidth. For example, if three incoming streams of bandwidth x are composed into a single stream, where there is no overlap of the various sources in the composite, the resulting stream will have an average bandwidth of $3x$. Hence, each station's incoming and outgoing link bandwidths are asymmetric. Some video encoding algorithms consist of different types of frames (e.g., MPEG), where some are reference frames which are encoded independent of any other frames, and predicted frames which are encoded based on reference frames. The decoding engine treats these frames differently, and cannot decode a predicted frame without the reference frames to which it applies. For these video streams, in order for the receiving decoder to be able to handle the composite stream as a single stream, the reference frames of each source need be synchronized to be mixed in the same composite frame, and switching should occur at reference frame boundaries. The H.261 video coding algorithm does not have this complication since the decoding

dependency is reduced to the macro block level (different macro blocks in a frame may either be of a predicted type or a reference type).

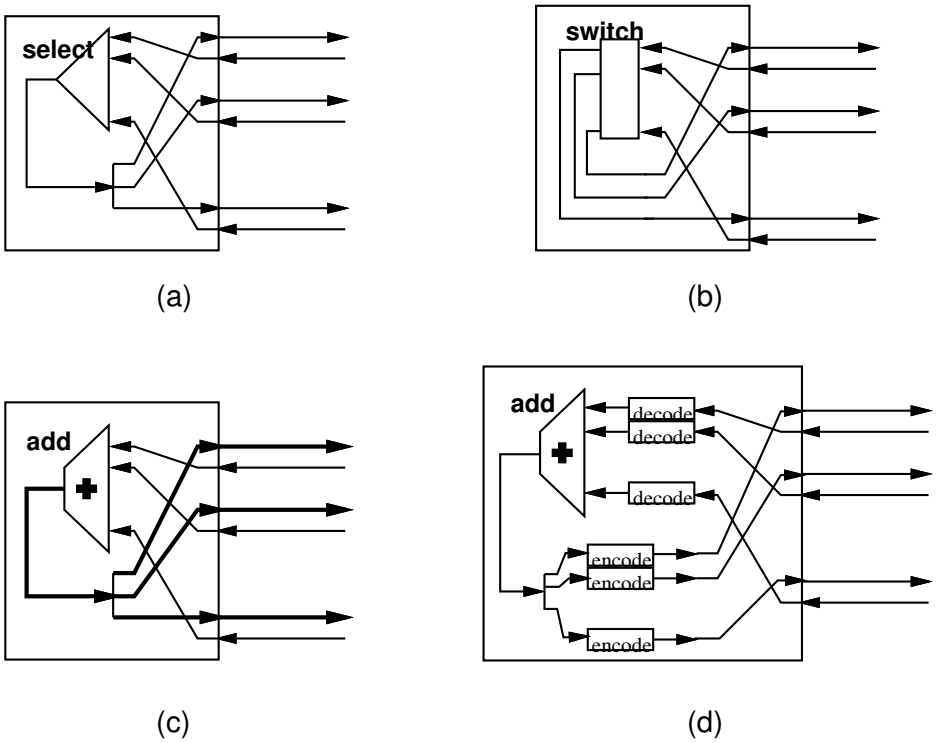

Figure 3. *Multipoint Control Unit Configurations. (a) Single video select and broadcast. (b) Destination video select. (c) Video composition in the compressed domain. (d) Video composition in the pixel domain.*

2.2.4 Uncompressed Composition

The final MCU configuration shown in Figure 3 (d) also performs a composition of the incoming video streams, but does so in the pixel domain. This requires that each incoming stream be decoded and that each outgoing stream be encoded. Separate encoders for each outgoing stream are depicted in the figure due to clock synchronization requirements between the sender and receiver. If the synchronization requirement can be solved otherwise, a single encoder might suffice. By composing the incoming streams in the pixel domain, the resulting stream bandwidth can be adjusted to match different link speeds. Another level of complexity may be added to the composition configurations by allowing each destination to select its own composition. The centralized approach is advantageous since the complexity of handling multiple streams can be confined within this single shared device and the network bandwidth requirements can potentially be reduced. However, it reduces the flexibility of the end-station control, and may increase the latency of the real-time traffic which is forced to traverse an intermediate hop between source and destination. Furthermore, a primary shortcoming of the centralized scheme is that it does not scale well. If one envisions large conferences (e.g., lectures) with thousands of users, it is not feasible for each participant to connect into a central server or MCU.

3. Packet-Based Approach

In a packet or cell-based network multiple virtual connections may be multiplexed over a given physical link. This packet multiplexing allows for efficient utilization of the link bandwidth, but also results in packet jitter

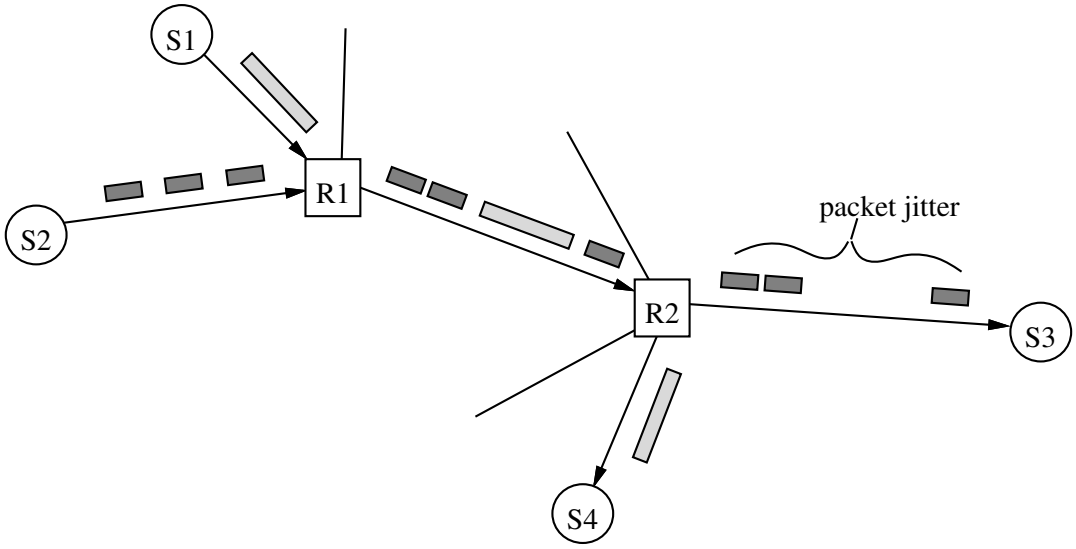

Figure 4. *Illustration of contention and jitter in a packet-based network where multiple streams of packets are interleaved along the same physical links. In this example a stream of packets being sent from station s2 to station s3 is contending at routers R1 and R2 with a stream of packets moving from s1 to s4. This results in jitter in the s2 to s3 stream.*

and, in the case of extreme contention, can cause packets to be dropped (see Figure 4). In today's Internet environment and in IP networks in general, quality-of-service (QoS) mechanisms have not yet been implemented to guarantee the delivery of packets with a negotiated level of QoS. For traditional data applications higher level protocols such as TCP/IP are used to provide an end-to-end guaranteed delivery of packets. TCP provides the guaranteed delivery through the use of retransmission schemes when packets are lost or excessively delayed. It also provides an inherent flow control since it adapts its transmission rate depending on the rate of acknowledgment receipts. For real-time applications involving audio and video, the end-to-end latency requirements are such that TCP's retransmission delays are too high to be useful. As a consequence, the UDP/IP protocol is used which has no end-to-end guarantees or inherent flow control. This places the burden of end-to-end synchronization and rate control on the videoconferencing application and requires that a level of robustness be built into the system to handle packet-loss and jitter.

A feature of the packet-based environment is the capability of routers and switches in the network to multicast a packet or cell to multiple outgoing links. The multiplexing and the multicasting capabilities may be leveraged to offer a distributed multipoint conferencing solution with advantages over a centralized approach. Hence, in a distributed multiway conference approach some of the MCU functions may be performed by the network while others can be performed within the end-stations.

The MCU component in the centralized approach performs several key functions to enable a multipoint conference. These functions can be categorized as control, multimedia distribution, and audio/video/data manipulation. Each of these functions are needed in a distributed environment as well, but where and how each of them are performed can vary. To better understand how some of the MCU functions can be performed in a distributed fashion within the network itself, we will examine the functions in more detail. We will first explore how the multicast capability in the Internet and ATM environments can be exploited to support a multipoint conference.

3.1 Multicast Support in the Internet and ATM Environments

Network multicast capability is essential for effective distributed implementations of the MCU functions described above. Hence, in order to explore how these MCU functions can be implemented in a distributed manner we first examine the multicast capabilities of the Internet and ATM network environments. Approaches for multicast support in the Internet and ATM environments, although

originally quite different, seem to be converging. Two very distinct multicast paradigms are sender versus receiver initiated tree constructions. In general, the receiver initiated approaches, first advocated by the IETF (e.g., DVMRP[17,18,19], MOSPF[20,21], CBT[22,23,24]), are better suited for densely populated groups while sender initiated approaches, as, for example, the scheme adopted by the ATM Forum [25], are better suited to smaller sparse groups.

In the receiver initiated approach described in [17-24, 26-27], a conference is established by obtaining a multicast address and advertising the time of the call along with the address to all participants. Connectivity is established amongst participants in the conference by creating a source-specific bi-directional multicast tree between all multicast routers using a broadcast.

End-stations send connect messages to the nearest multicast router to subscribe to a particular multicast address. A router is pruned from the tree only if no end-stations have registered with the router to connect to the conference multicast address. Every so often (after a predefined time-out period) all routers are reconnected via a broadcast and then pruned again depending on each router's end-station registration. This receiver initiated multicast solution is ideal for large and densely connected multiparty conference models. It scales well to support many users, incurs no additional overhead for adding or deleting participants, and is robust in the event of failed routers or participating end-stations. It is not well suited, however, to sparsely connected end-stations and offers little, if any, means of controlling participation. The PIM architecture has been proposed [26,27] to address both the case of sparse groups as well as densely populated groups. It supports two modes of operation, PIM SM and PIM DM, for sparse mode and dense mode, respectively.

A sender initiated approach, as adopted by the ATM Forum [25], defines both bi-directional or unidirectional multicast trees with the sender at the root and the recipients at the leaves. In this model a station must sequentially add each station it wishes to communicate with to its multicast tree. This sender initiated approach is well suited to small group conferences, where participants each know about each other. It does not scale well for larger conferences since the overhead of adding and deleting participants is incurred by each end-station in the conference. The shortcomings of the sender initiated approach have motivated a recent contribution to the ATM Forum describing a variety of alternative means for the support of ATM multicast [28] services some of which are closely modeled after the IETF schemes.

3.2 Videoconferencing Multicast Model

The ideal multicast model for a distributed multiparty conference, in our opinion, would allow each receiver to dynamically select which of the other participants it chooses to view. This requires a flexible receiver controlled multicast tree formulation. A separate multicast tree would carry audio in order for all participants to be heard. Finally, another multicast tree would distribute the data.

Distributed multiway conferencing schemes vary depending on the types of functionality they are designed to support. Let us consider how a distributed version of the various multiway conference capabilities provided by the four different MCU configurations described in Figure 3 might be implemented.

Configuration (a) requires that all participants be heard, but only that a single participant's video be viewed at a time. To achieve this, a single multicast tree can be used to carry audio as well as a single multicast tree for video. However, a special arbitration protocol is required to select which participant's video to transmit at a given instant. A voice driven activation scheme could be implemented for participants to dynamically activate or halt their video transmission. Additional arbitration mechanisms may be required to prevent two or more participants from transmitting video simultaneously in the event that they possibly begin speaking simultaneously.

Configurations (b) and (c) are similar in terms of network requirements, but differ in terms of end-station complexity. In both schemes, separate multicast trees are required for each participant's video. To receive a particular video source a participant must subscribe to the appropriate multicast tree. In order to be able to dynamically switch between sources, the process of joining and leaving multicast trees needs to be efficient and fast. As far as the end-station complexity is concerned, in (b) each station only needs to be

able to handle a single incoming video stream. However, in (c) each end-station must be capable of accepting multiple incoming streams and processing them accordingly to create a composite view.

In a distributed approach, based on the network multicast capabilities described in the previous section, each end-station receives the video and audio streams from all, or some, of the participating end-station sources in the conference. Depending on the end-station complexity, it may then display a single incoming video stream or create a composite of multiple incoming streams for display. This approach allows the potential for more flexibility and control at each end-station and minimizes the latency of video and audio streams between source and destination. However, it requires additional processing capability at the end-station, and potentially greater bandwidth requirements from the network depending on the desired richness in functionality.

3.3 Separation of Control and Multimedia Streams

The first aspect of a distributed multiway conference model that may differ from the centralized MCU approach is the separation of control streams from multimedia streams (e.g., video, audio, and data). In fact, one solution might maintain a centralized approach for control via a designated server node in the network while implementing a distributed approach for the multimedia streams. A centralized control approach would differ little from the MCU-based control, but require that the MCU establish appropriate multicast connections for the multimedia streams.

The centralized control approach would suffer from the same scalability shortcomings as the MCU solution. In a centralized control structure, operations such as video selection or caller acceptance can easily be handled. For these functions to be performed in a distributed way requires more sophisticated algorithms, particularly to be performed in a timely and robust manner. An interesting topic is that of secure communication. Further exploration is required to investigate how multimedia streams may be protected and encrypted in a multiway conference. If the multiway scheme is centralized and the MCU needs to decompress the audio or video stream, then the MCU needs to be part of the secure system. If, on the other hand, the video and audio streams are passed directly between senders and receivers then they can be sent using conventional encryption techniques.

3.4 Synchronization

There is no provision of a global clock in today's packet-switched networks. This introduces a potential clock mismatch problem between the source and destination stations of a real- time video conferencing system. When dealing with constant bit-rate streams as the H.320 standard defines, a minor clock mismatch will eventually cause a buffer overflow or underflow. Another consequence of packet-based networks is jitter. Playout buffers may be incorporated at the receiver to absorb jitter and adjust clock mismatches, but these may also contribute considerably to the end-to-end delay. For example, if a system has a 1 Kbyte playout buffer and the system generates audio data at an 8 kHz sampling rate with 1 byte per sample at the source. Then, although the buffer size is relatively small, it already corresponds to 128 ms of delay time when it is built up from empty to full.

A typical crystal oscillator has a 10**(-4) frequency tolerance. Assume that the receiver has a crystal oscillator with a clock rate which is 10**(-4) slower than the transmitter. In this case, for every second there will be 0.80 bytes more audio data generated at the transmitter than consumed at the receiver. The buffer will overflow from empty within 1280 seconds (about 22 minutes), and the end-to-end delay also increases as the buffer starts to fill. This example indicates that in 22 minutes, the end to end delay will increase by 128 ms due to the clock mismatch only. Given that the target end-to-end delay for real-time video conferencing is on the order of 300 ms, this added delay is significant. Furthermore, since a major part of the 300 ms delay budget is expected to be spent in the packet network, the end station delay should be minimized.

There are various network or higher layer protocols for dealing with the real time clock issues. For example, the Internet draft real-time transport protocol (RTP[29]) is proposed to provide end-to-end delivery services for data with real-time characteristics. RTP is a higher layer protocol which specifies that each packet should carry certain information such as time stamps and sequence numbers for use in network nodes (e.g., routers, bridges or translators) to provide real-time services. However, RTP itself

does not provide any mechanisms to provide quality of services guarantees or real time delivery. Another such proposal was the Internet Stream Protocol (ST-II [30]) at the same layer as IP. ST-II is designed to use the pre-allocation of resources to allow packets to be forwarded with low delay and low loss due to congestion. However, these schemes do not consider the effect of clock mismatch between the source and receiver (playback) stations.

The clock mismatch issue was most recently addressed in the MPEG-II [31] transport layer The MPEG-II transport layer periodically inserts a Program Clock Reference (PCR) signal within the MPEG II stream. The decoder has to slave the local clock to the source PCR in order to achieve source destination clock synchronization. It is noted, however, that this scheme will not work properly in a packet network environment which contains jitters nor in the variable bit rate (VBR) encoding environment. The low end to end delay requirement will also put an upper bound on the size of the receiver buffer. Unfortunately, this master/slave scheme fails completely in the multiparty environment.

3.5 Audio and Video Stream Manipulation

Although not all video signals need be viewed simultaneously for a videoconference to be effective, all audio inputs must be heard by each participant. For a large or even moderate size conference it is not practical to mix all the audio signals at each end-station. Hence, for a multiway conference an effective way of supporting a larger number of participants is through the use of silence detection. By using silence detection mechanisms at each source and only transmitting audio during active periods, only a finite number of audio signals need to be mixed at any given instant. When more than this finite number are active simultaneously, the mixed audio signal is not intelligible anyway.

When going through an MCU and composing multiple video streams there are 2 options. The first option is to decode each incoming stream, compose the selected stream, and re-compress the composite. This approach is necessary if the bandwidth of the composed stream is to equal the bandwidth of a single incoming stream. The decompression and recompression adds delay to your end-to-end latency, and to allow customized composition for each recipient is very expensive. The second option is to create a composite of multiple streams in the compressed domain. This is advantageous in terms of delay, but by combining, say 4 QCIF streams to create a CIF stream, your outgoing stream is 4 times the bandwidth of each incoming stream. Note that if these composites are customized for each recipient then you could potentially have N composite streams.

4. Multimedia Multiparty Teleconferencing -a Distributed Solution

Due to its multi-disciplinary nature, validation and evaluation based on a working test-bed are very important aspects in the research and development of a multimedia desktop conferencing system. Consequently, the Multimedia Networking Group at IBM T. J. Watson Research Center is engaged in the development of a comprehensive multimedia multiparty teleconference (MMT) prototype. A primary objective of the MMT project is to investigate various aspects of the distributed multiparty conference model. In particular, we present a solution which takes full advantage of the multicast capabilities of a packet-based network environment. We then present an efficient way of handling the mixing of multiple audio streams and multiple video streams within each end-station. Two other important issues which the MMT solution addresses is end-to-end synchronization of the audio and video CODECs and the use of an adaptive rate control scheme to adjust to the varying network bandwidth conditions in a packet-based environment.

4.1 MMT structure

The current MMT prototype is a JPEG-based system designed for distributed multiparty conferencing over a packet switched network. The hardware configuration of the MMT system (see Figure 5) consists of: a video/audio capture and display card (VAC), a CODEC and DSP card (MCD), and an optional network communication adapter. The VAC card is responsible for digitizing the incoming video/audio signal and

the reverse process of video/audio playback. The VAC card will accept NTSC as an input signal and generate an NTSC signal as the output.

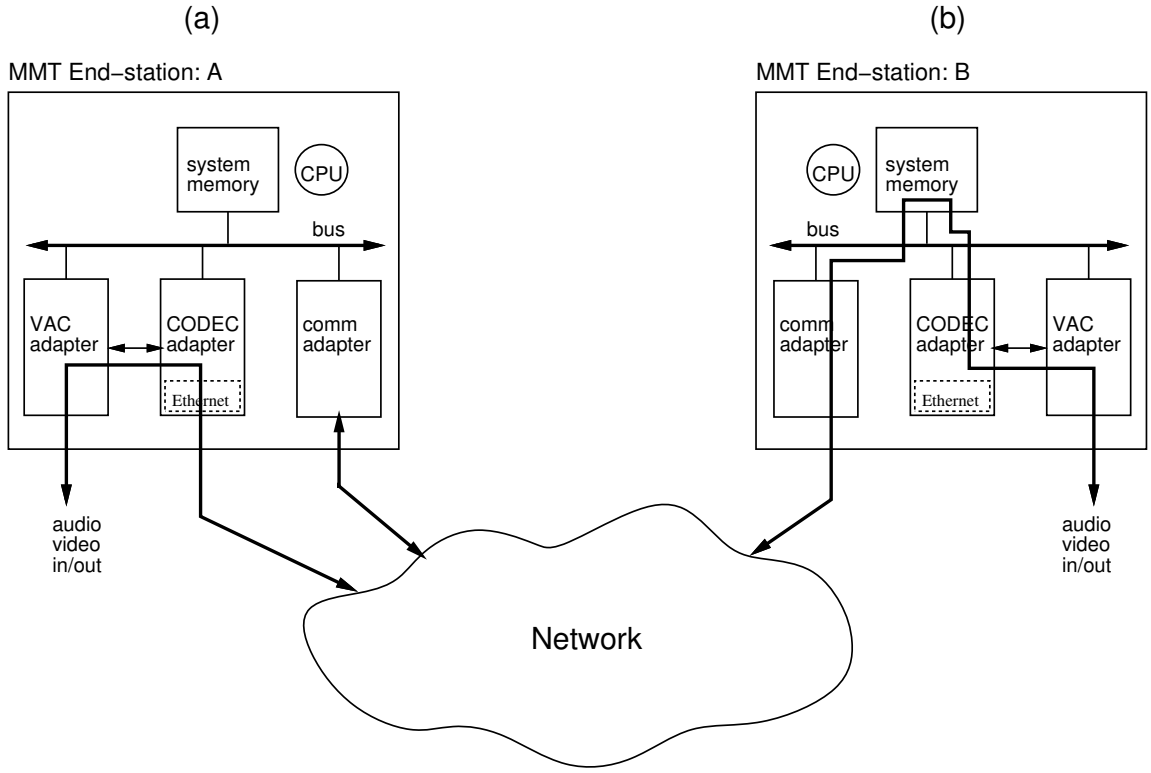

Figure 5. *MMT system block diagram with alternative communication modes:*
(a) Via the on-board Ethernet. (b) Via the microchannel bus through a separate network adapter.

In the MMT system audio, video, and data packets are sent as separate streams. This allows receivers to elect to receive only an audio or data stream from a particular sender rather than be forced to consume all three. This is particularly useful for stations with no video processing capability. Silence detection is implemented such that there is no audio packet transmission during silence periods. The VAC card is also capable of removing the annoying frame segmentation effect, known as shearing effect, in the playback.

The MCD card implements the video compression and decompression. The DSP algorithm implements audio mixing and video composition such that the end user can choose to hear or view any arbitrary conference participant(s). The DSP code also implements an algorithm for concealing the effect of clock mismatch, lip synchronization, and packet loss. There is a direct interface between the VAC and MCD adapter. This enables the video and audio data to bypass the microchannel bus and frees CPU power for data sharing or other applications. There is also an on-board 10 base-T Ethernet communication system in the MCD.

This on-board 10 base-T interface enables MMT to connect directly to the LAN without the intervention of the CPU. In addition to the on-board 10base-T interface, the compressed data can be sent up to the PC system memory via the microchannel bus. This enables the system to be configured with any UDP/IP supported communication adapter.

4.2 Network Multicast Support

Although multicast is a well-know concept, it has not been widely implemented in existing networks. Local area networks provide a hardware multicast capability but it is not available at the higher protocol layers. Consequently, to achieve the desired multicast capability several approaches were experimented with using the MMT system:

1. A direct interface to the ORBIT/PARIS [32] gigabit network. ORBIT/PARIS is an experimental gigabit network and is one of the five national gigabit testbeds coordinated by CNRI. ORBIT/PARIS implements a proprietary network multicast function. As such, each MMT station sends one multicast type packet and receives all appropriate multicast packets from other participants.

2. Connection to an Ethernet switch hub. This hub implements the Distance Vector Multicast Routing Protocol (DVMRP). By establishing a tunnel between two hubs, the switch hubs are able to provide a multicast IP network superimposed on top of the Internet.

3. Packet replication over the traditional Internet. Since there is no multicast support in this environment the MMT stations were made to simulate the multicast by sending out multiple replicated packets for many point-to-point sessions for each individual destination

In the current MMT prototype the multicast connections are setup using predetermined multicast addresses. This is one area that is currently under further investigation. Network directory services are required which will more easily assign addresses and provide confidentiality and security.

4.3 Video Composition in the Compressed Domain

The network multicast capability allows each station to effectively receive multiple audio and video streams simultaneously. To process (decompress and render) these multiple streams in real-time is computationally intensive and potentially prohibitively expensive. Hence, one problem solved by the MMT system is to provide an efficient video mixing and playback algorithm so that a multimedia personal workstation can receive streams of packet videos from different sources and display them as a set of user defined overlaid windows.

A high level view of the video mixing problem is illustrated in Figure 6. Videos are generated by multiple independent sources such as the three sources, i.e., A, B, and C, in the figure. At their individual source, video is first compressed according to JPEG format, packetized, and then transmitted into a high speed packet switched network. At the receiver, video packets coming from different sources arbitrarily interleave in their arrivals. The task is to design a good algorithm to process videos, such as A, B, and C, so that they can be displayed as a set of user defined windows. Windows may overlay on top of each other, such as the case of video B to video A. Videos are generated, compressed, and transmitted independently from different sources. They may have different sizes and different frame rates. For example, video A may be 320 by 240 in NTSC, while video B may be 480 by 300 in PAL. The assumption is that they are compressed based on the same JPEG format, and a real time JPEG decoder is available. The decoder is capable of decoding videos of the maximum display dimension.
To illustrate the points of interest, we will discuss the limitations and complexity of a straightforward approach, which is also depicted in Figure 6. The approach is to have one compressed data queue for each arriving stream of compressed packet videos. These packets, at an appropriate time, are fed to the JPEG decoder for decompression. The decompressed pixel data from the decoder are stored in a pixel data buffer before the final mixing. Then, a mixer is in charge of taking data from the pixel data buffers, one for each source, and write them into a display frame buffer, which is the bitmap (or pixelmap) of the final display.

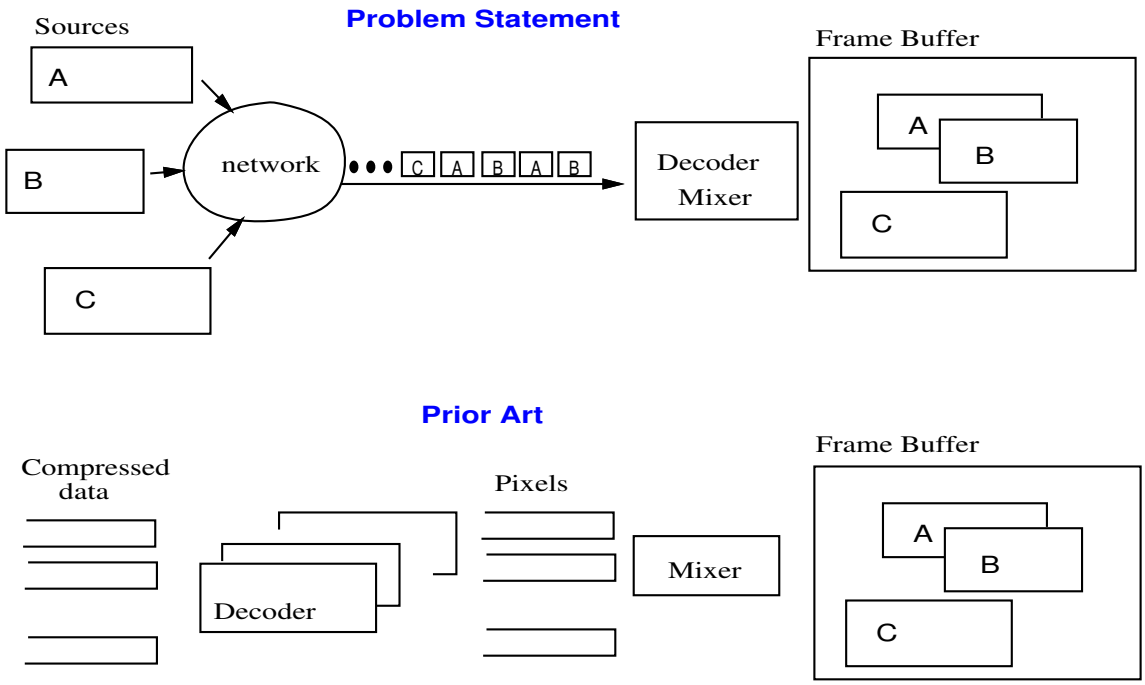

Figure 6. *Illustration of the video mixing scenario.*

In the above approach, the decompression capability in the decoder must be able to handle, in real time, a number of pixels equivalent to the total number of pixels from all video windows, which can be as large as n times of the display area, where n is the number of video windows. This is quite expensive as it would most likely require a bank of decoders running in parallel. Moreover, it is necessary to have enough pixel buffers to hold at least one frame for each video window. This means a lot of buffering when there are a large number of video windows, even when most of the pixels are covered due to overlaying.

A more desirable approach, if possible, is to do the mixing directly with the compressed data. In this approach, decompressed pixel data is written directly to the frame buffer for display. Consequently, there is no need for the large amount of pixel buffers and real time video rate circuitry, as required in the previous approach.

The concept of mixing is essentially combining pixel data of several video windows. If there is no clear association between a stream of compressed data with the pixel data in the corresponding video window, it seems unlikely that mixing can be done in the compressed domain. In this paper, however, we will demonstrate that mixing in the JPEG compressed data domain is possible, and propose an efficient design and implementation of such an approach.

4.3.2 JPEG Base-Line DCT System

In this section we will describe the JPEG base-line DCT system to provide the necessary background for introducing the new mixing technique in the next section. Two approaches are defined in JPEG: a spatial prediction algorithm and a discrete cosine transform (DCT). In the DCT approach, a "baseline system" is defined, which is a subset of features that are sufficient to guarantee a reasonable level of performance. The baseline system is required for all JPEG DCT systems.

There are three steps in the baseline DCT system. The first step is to transform the image pixels to

their spatial frequency signals by DCT. In this step a video frame is divided into 8 pixel by 8 pixel blocks.
The transformation is done separately for each block. The resulting signals are represented as DCT coefficients. The second step is to quantize the DCT coefficients to reduce the data rate. The quantization matrix is chosen so that higher frequency DCT coefficients, which are usually smaller and less critical to video quality, become zeros. The quantized DC and AC coefficients are represented in different coding format.
The DC coefficients are coded in differential pulse code modulation (DPCM), because they often vary only slightly between two consecutive blocks. The AC coefficients are coded using the run-length technique because there usually are runs of consecutive zero value quantized coefficients. This step, which is an information lossy process, accounts for the major part of data reduction.

The final step is to do Huffman coding to further reduce the data rate by minimizing the entropy in the DPCM and the run-length codes. The reduction in data rate comes from using shorter code words for more frequently occurring symbols. The amount of compression is dependent upon the source content and the required quality in the reproduced video. For example, 0.75 bit/pixel is sufficient for excellent quality reproduction of most natural color scenes; 1.5 bit/pixel is needed to achieve indistinguishable reproduction quality for the same types of scenes. This means a compression ratio, i.e., the ratio of the original data rate to the compressed data, of 32 and 16, respectively, assuming the original data rate is 24 bit/pixel.

A set of marker codes is defined in JPEG to delimit the compressed data stream. Each marker code is a two byte-aligned 8 bit integer. The byte aligning is designed so that marker codes in a data stream can easily be identified. Several marker codes are important to the new technique and are described below:

SOI (start of image): starts a data stream.
SOF (start of frame): starts a frame.
SOS (start of scan) : starts a scan.

A frame consists of several scans. One scan consists of several coding intervals. A coding interval is an integer multiple of the minimum data unit (MDU), which is two consecutive 8 pixel by 8 pixel blocks or a 16 pixel by 8 pixel block.

RSC (resynchronization): terminates the DPCM dependence between DC coefficients.
EOI (end of image): terminates a data stream.

In JPEG, an image frame can be any size up to 65535 pixels by 65535 pixels. For simplicity and without loss of generality, we will use VGA, which is 640 pixels by 480 pixels, for the display screen resolution in our discussion.

4.3.3 Feasibility of Mixing in JPEG Compressed Domain

In general mixing of multiple videos can be done only in the pixel domain, because of the lack of association between the compressed data and the pixels, particularly in a variable bit rate coding scheme. For JPEG however, we have found that mixing directly with compressed data is possible. The way to do this is to use RSC's to delimit and map the coding intervals to blocks of pixels. A coding interval can be as small as an MDU, which is a 16 pixel by 8 pixel block. This granularity, when compared to the size of a typical screen, such as VGA, should be fine enough for window overlaying of most applications.

A simple system architecture that can use this new mixing technique is as follows. All sources use the same size coding interval, say a coding interval is an MDU. Each video window is partitioned into a grid of CI's, which are labeled sequentially from left to right and from top to bottom. The position labels of

the CI's carried in a packet are encoded in the packet header. In this way, if the receiver knows the dimension of a video window and its coding interval size, it may use the position label to map any coding interval to its corresponding block of pixels.

At the receiver, the display screen is partitioned into a grid of CI's, which are also labeled sequentially from left to right and from top to bottom. For example, if the coding interval is an MDU, then a VGA display screen is partitioned into 2400 coding intervals, as depicted in figure 7.

4.3.4 Video Mixing

We assume that compressed data is received and processed in such a way that CI's and their position labels are stored in separate queues according to their sources, one queue for each source. We also assume that a separate playback module, which is responsible for handling packet loss, corruption, and delay jitter, ensures that the compressed data of a complete frame is ready in each queue for mixing.
An "Overlay Table (OT)" is used to specify the overlay configuration of video windows. The OT has one entry for each CI on the display screen, as depicted in figure 7. For the case of a VGA resolution display, the OT has 2400 entries. Each entry maps one CI on the display screen to a specific CI of a specific video window. This information is derived from a user's definition of the positions and overlay placement of video windows. A blank CI is represented by a special symbol in the source field.

Assume one composite frame is created every P seconds. To create a composite frame, the composer goes through every entry in OT and either reads out the specified CI from the specified queue, or generates a blank CI. In addition, the composer also needs to embed in the new composite compressed data stream the necessary marker codes such as RSC and SOS, quantization table specification, ..., etc, to satisfy the JPEG format. The JPEG decoder then reads in the composite compressed data stream, decompresses it, and outputs the pixels in a form that is suitable for display.

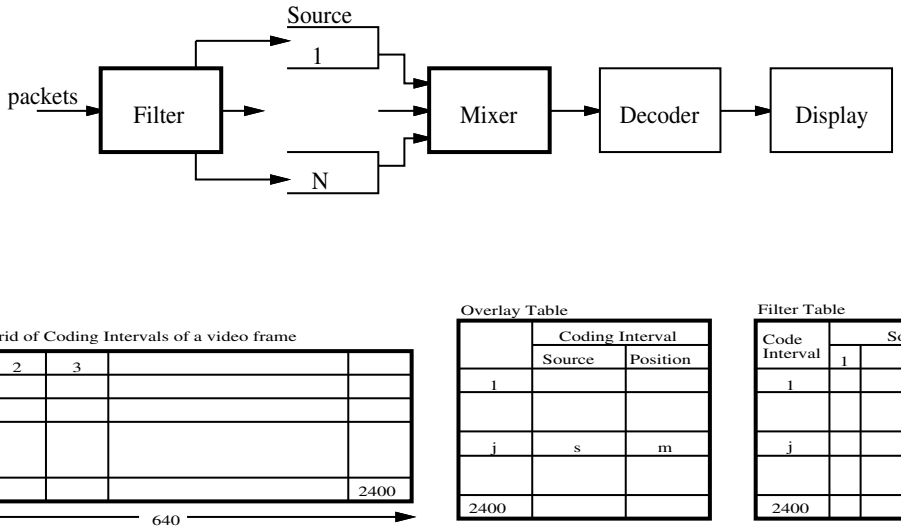

Figure 7. *The structure of video mixing in the compressed domain.*

Let Hs and Vs be the number of pixels in the horizontal and vertical dimensions, respectively, of the display, and Hi and Vi be the horizontal and vertical dimensions of the video window i.
In this structure, there is no need for pixel buffers. In addition the decompression is performed only for one screen worth of pixels, i.e., Hs Vs, in one frame time. Hence, any typical JPEG decoder, which most likely was implemented without any concern of mixing multiple video windows, is sufficient.

4.3.5 Video Filtering

In the previous section, we assume that the compressed data of a complete frame is always ready in each queue for mixing. This implies that it requires more buffers for compressed data than it requires to display. This buffer requirement can be reduced by observing that a CI need not be enqueued if it is covered by another CI of some video window. With appropriate packet filtering, the buffer requirement is reduced to the amount needed to hold a screenful of pixels, regardless of the number of video windows and their dimensions.

Figure 7 depicts the packet filtering structure. A Filtering Table (FT) is used to guide the filter in discarding or buffering an arriving CI. FT has one entry for each CI in each video window. For the VGA resolution display example and the common coding interval size of an MDU, the FT has 2400 entries, because each video window can be as large as a VGA screen. Each entry has N flags, one for each source. A flag is set if the corresponding CI will be displayed; otherwise, it is overlaid by other video windows. Therefore, a CI is stored only if its flag is set.
With the filtering, the structure needs only to store up to a display screen worth of CI's. In VGA, for example, the buffer requirement is 921 Kbytes divided by the compression ratio. If the ratio is 20, then the buffer requirement is 46 Kbytes, which is quite small.

4.4 Clock and Lip Synchronization

The MMT system employs a novel, yet simple, scheme to solve the clock mismatch and lip synchronization problems. The basic clock mismatch problem exists in any packet switch environment without a global clock to which each station can synchronize. Each station effectively has an independent clock. For point-to-point connections it is possible to implement adaptive clocking techniques whereby the receiver slaves to the sender clock ,or vice-versa, based on the rate of the incoming data. This solution is not suitable for multiparty scenarios where multiple senders and/or receivers need to synchronize to each other. In our opinion, the lip synchronization scope should be limited to the video/audio pair arriving from the same source. Since the audio and video streams are independent, they must be synchronized at the receiver in order to render the corresponding audio and video frames simultaneously. We begin by describing the MMT point-to-point synchronization mechanism first.

Since the MMT system does not transmit audio packets during silence periods, the audio sequence is not a continuous flow but a sequence of talk spurts separated by silence. Hence, using an audio master and video slave strategy, we are able to solve the lip synchronization problem. In this scheme, audio packets are played back freely at the pace of the receiver audio clock. The audio clock mismatch between the sender and receiver is absorbed automatically in the silence intervals.

Timestamps in the audio and video streams are used to force the video frames to synchronize with the closest audio segment only at the beginning of each talk spurt by either discarding some delayed video frames or reusing previous frames until the audio and video timestamps match. This scheme does not require the playback station to lock into the capture station clock rate as in the various master/slave adaptive clocking techniques. The playback station will free run at its own clock rate for the duration of one talk spurt and lip synchronization will be attained at the beginning of each talk spurt. The justification for the performance of this scheme is based on three observations: the average length of the talk spurt is very short (about 1.32 seconds) [33], the average length of a silence period is long (about 1.82 seconds) [33] compared to the talk spurt, and the clock mismatch between the capture and the playback is small. As a result, lip synchronization does not drift by much in such a short interval. This lip synchronization scheme proved to work well in packet-based network environments where packet jitter, loss and clock mismatch occurs.

The same scheme is extended to the multi-point conferencing environment, where every capture station will have its own capture clock rate. The receiver station can not lock into every one of them as the master/slave adaptive clock techniques prescribe. Instead, MMT synchronizes the video and voice

streams which belong to the same sending station individually. The timing relation between individual media sequences of each sending station depends completely on the arrival time of the independent media sequences.

An important aspect of the packet-switched environment is packet-jitter and packet-loss. MMT addresses this through the use of timestamps and inherent robustness within the JPEG video and raw (PCM 16ksample/second and 16bit/sample) or G.711 (64 kbps) audio streams. A fixed length audio playout buffer is automatically filled by incoming audio samples which are placed within the buffer depending on their timestamp value. Lost or excessively delayed audio samples are replaced with the previous audio sample to reduce the resulting audio noise perceived by the user.

5. Rate Control

As mentioned earlier, the UDP/IP protocol is used to transmit real-time audio and video over the Internet. This is due to the low-delay requirements and also to make use of the multicast facilities. Unlike TCP, UDP does not have a notion of flow control. Consequently, UDP multimedia traffic can easily congest the network if the application program does not implement rate control at the source.

A fair amount of research has gone into the area of adapting the video encoder to the available network bandwidth. All of the proposed schemes invariably adjust the parameters in the encoder to get the desired data rate. Kanakia, Mishra, and Reibman's [34] goal is to minimize the effect of network traffic overload on the quality of the resulting image. They proposed to match the source's sending rate to that of the bottleneck in the network, wherever that may be. Their scheme requires that all nodes in a network participate in collecting and reporting their network states. Bolot and Turletti [35] uses a loss-based feedback scheme to adapt source's output to the availability of the Internet. For small sessions (<10 participants), feedback is in the form of a NACK sent back to the source. For large sessions (>10 participants), feedback is in the form of a QoS message with a probability which is sent periodically by the receiver. Their encoder is H.261 compliant, and they adjust the quantization factor, frame rate, and interframe compression threshold to adjust the source's output to a desired rate. Chakrabarti and Wang [36] use a proprietary encoding system. They adjust the threshold dictating the spatial resolution in order to control the amount of data required to encode a frame. They are concerned with stabilizing the frame rate and maximizing throughput. Noting that the receiver's display rate is what drives the system, they move the rate adaptation to the sink. The scheme uses a leaky bucket for frames at the receiver along with a timer to drive/calculate the current rate. Sakatani [37] proposes to detect congestion by (1) measuring the number of collision/retries at CSMA/CD level and by (2) measuring the round-trip delay (delay-based feedback). If each of the above quantities is above a certain threshold, then he notes that congestion has occurred. Once congestion is detected, he implements a "slow start" algorithm to reduce congestion with only two levels of bit rates. His encoder conforms to the H.261 standard, and he changes the quantization factor and frame rate to achieve the desired output rate. He does not show the effect of his work on quality. Finally, Panda and El Zarki [38] examine the impact of several compression parameters to be used in controlling the traffic characteristics of an MPEG source.

We will examine by experiment a simple method for source rate control. This method does not require any changes to the encoder's parameters (as a result each frame maintains the same quality factor). Rate is controlled at the source by adjusting the frame rate to the input of the encoder. The target output data rate is fixed by the user or dynamically adjusted by feedback from the network. Experimental results show the effectiveness of the method.

5.1 Experimental Setup

Our rate control mechanism and adaptive scheme is implemented with the MMT system. Each MMT station contains a program, running as a user process, to initiate and control the MMT hardware. A software process is used to collect MMT compressed video and send it over a network using UDP/IP to another MMT station. The user process can initialize MMT with several parameters at startup, including window size, quantization factor, scale factor, and frame rate. While it is possible to adjust all these parameters during run-time, all but the frame rate requires time consuming reinitialization of the hardware system. For example, to change the quantization factor, the source has to notify the receiver that it is changing the quantization factor. The receiver then has to reload the quantization table. If the message that the source is about to change its quantization factor is lost, then the ensuing data becomes garbage to the receiver. Thus in our environment, it is only feasible to change the frame rate during runtime.

The controlling program communicates the desired frame rate to MMT. MMT then uses a table to determine which frames to drop while optimizing the smoothness of the resulting stream. This is achieved using a set of bit masks which correspond to frame rates between 0 and 30 per second, and a modulo 30 circular bit index. At 33 millisecond intervals, the system checks the specified frame rate from the program, pulls out an appropriate bit mask, and performs a bitwise ANDing of the index with the bit mask. If the result is one, then the frame is compressed, otherwise the frame is dropped. For our experiments the video was sent at a quantization factor of 20, and the frame size was set at 320x240.

Our network consists of an Ethernet segment connecting the source MMT station to a bridge. The bridge is connected to the IBM 16 Mbps token ring network that connects to the receiving MMT station.

5.2 Source Rate Control

In a bandwidth limited environment, large data bursts can cause severe congestion, degrading the video quality seen at the receiver. However, the variability of the video data makes it hard for the system to allocate buffer space or CPU cycles. If we allocate too much resources, then the system goes under-utilized. If we allocate too little resources, then the performance is compromised.

Optimizing the quality at the receiver involves controlling the output data rate at the source to avoid congestion which leads to delay and loss. The source data rate is affected by the content in each video frame and its integrity is affected by the network on the way to the receiver. The first step is to control the data rate at the source to a given bandwidth. As we will show later, this step is significant in reducing the congestion. Once we have successfully controlled the data rate, we should match it to the network's available bandwidth, which we can determine using feedback.

Most video compression algorithms will produce compressed streams which vary in bit-rate depending on the variations in the video content. Scenes which contain more motion and more complex objects in terms of texture and detail will require more bits than low-motion scenes. To prevent the source from swamping the network when the encoder generates a higher data rate, we need to limit the source output to be below a given target bandwidth. There are many different approaches to rate control. For example, a communication adapter traffic shaper may use a leaky bucket flow control scheme which buffers data before transmitting it into the network to prevent a given connection from violating its negotiated traffic parameters. Although the traffic shaper will help insure that the connection conforms, it does so blindly without knowledge of frame boundaries. A traffic shaper may not only cause delays, it may also be forced to drop packets arbitrarily. For our JPEG video implementation, if any portion of a frame is lost, then the whole frame is discarded. So, if there is a small amount of packet loss for each frame, then the receiver's frame rate would effectively drop to zero. If the data rate is controlled at the encoder, then more intelligent steps can be taken to reduce the bandwidth and discard packets systematically (al packets of the same frame) or avoid compressing a frame altogether. This preserves bandwidth for the frames that have been encoded, and improves the receivers effective frame rate.

The MMT hardware drops frames selectively given a target frame rate. To accommodate for differing frame sizes due to video content, we calculate frame rate for each encoded frame as follows:

frame rate = target bandwidth / current frame size.

The target bandwidth is a number given by the user or system, or it may be determined dynamically. In the case of a change in frame size (in terms of height and width) our scheme is able to adapt to that change and reduce or increase the frame rate, thus keeping the output data rate constant.

We calculate a new frame rate when the current frame size is different from the previous frame size. To limit oscillation and constant evaluation of the frame rate, a running average of the data rate for N frames is calculated and compared to a predetermined threshold when evaluating the current data rate.

diff = average of (frame(i) to frame(i-N)) - average of (frame(i-1) to frame(i-N-1));

if (|diff| > threshold)

compute new frame rate

5.3 Performance Results

Our initial experiment using MMT's JPEG video CODEC, reveals that the data rate varies considerably over time. Our test video consists of one minute of white screen, one minute of a football game, and one minute of a talking head. These three samples represent different types of content with very different compressed data rate profiles.

Figure 8 shows the variation in frame size of our test video over time. Note that the size ranges from about 7 Kilobytes to 28 Kilobytes. At one point, the size changes from 12 Kilobytes to over 24 Kilobytes in just a few frames.

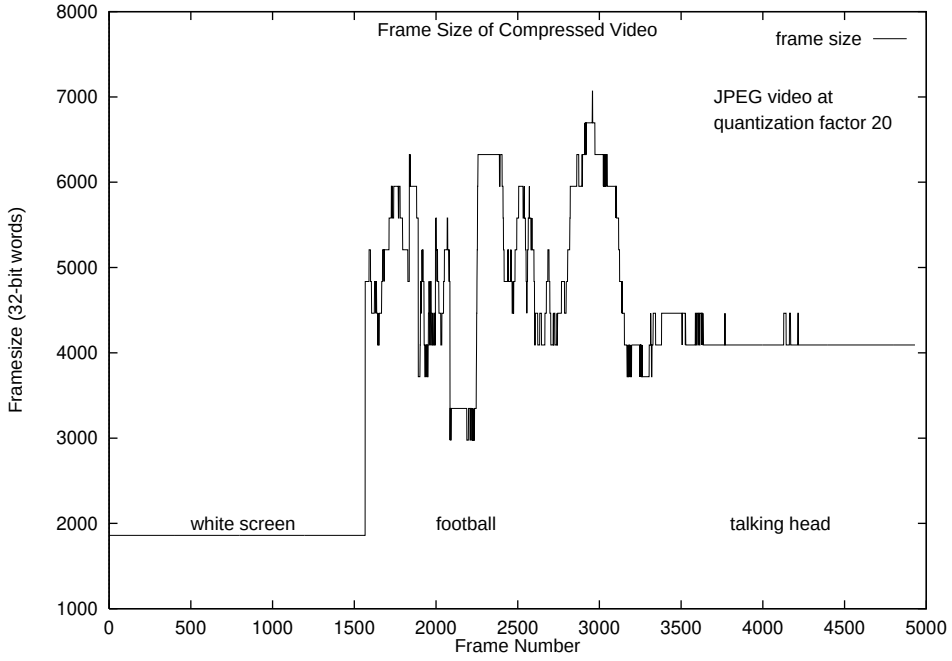

Figure 8. *Frame size of the test video sequence.*

Our experiment reveals that if the source is not controlled, its output data rate would congest the network. We ran MMT with our test video sequence over a packet-switched network. Note that due to congestion, the bandwidth seen at the receiver is less than that at the source. When the source's bandwidth is large and causes congestion, data is lost and the receiver sees significantly less effective bandwidth. Sometimes the effective

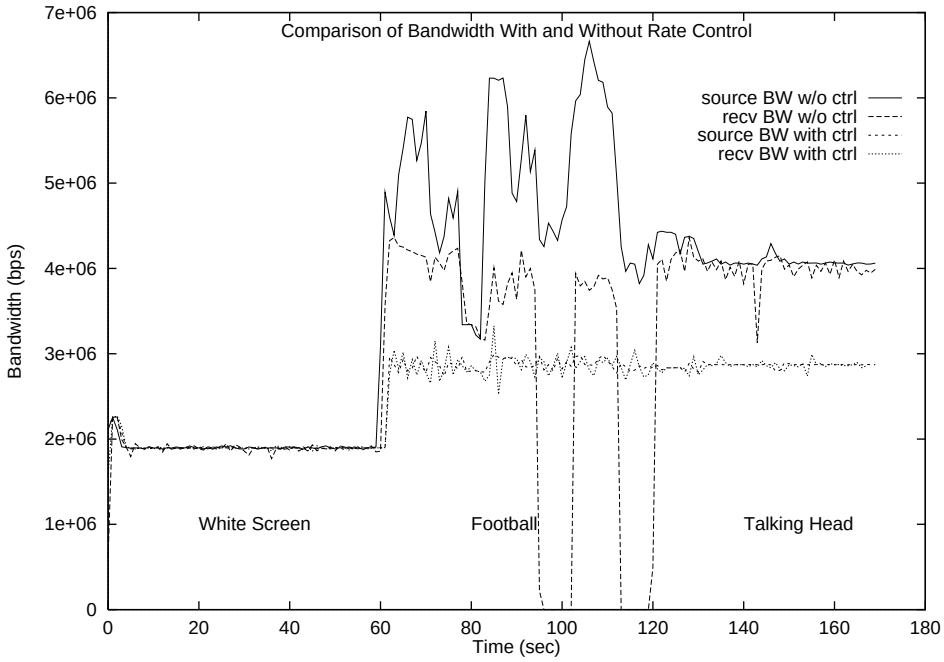

Figure 9. *Comparison of bandwidth with/without the rate control.*

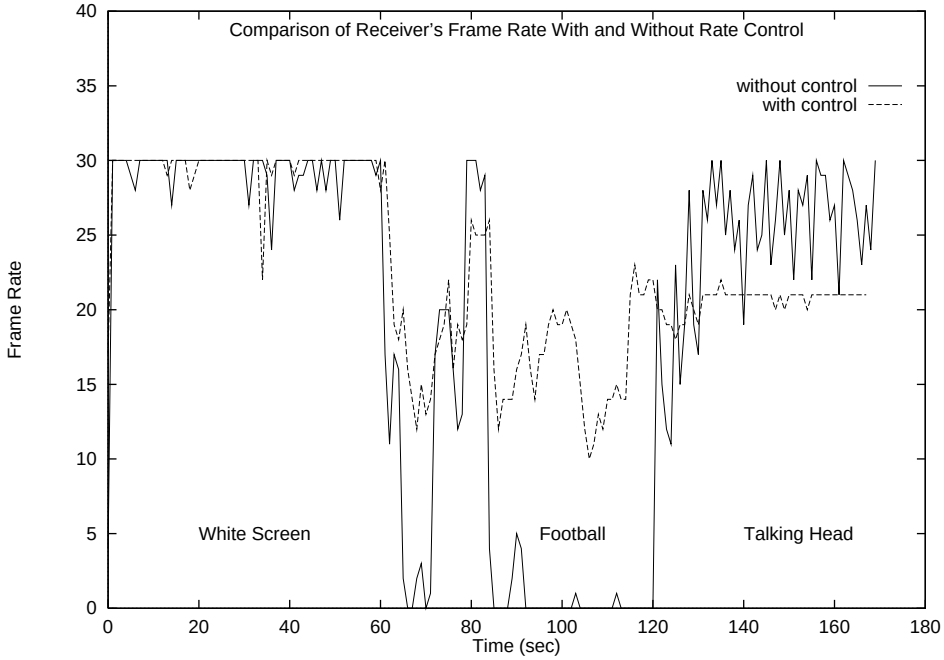

Figure 10. *Comparison of receiver's frame rate with/without rate control.*

bandwidth drops to zero. Our first result shows the dramatic impact of rate control on video over a network. Figure 9 shows the bandwidth generated at the source and bandwidth seen at the receiver.

Note that for the uncontrolled version, the source output varies dramatically over time, ranging from 2 Mbps to well over 6 Mbps. However, the receiver never sees more than 4 Mbps. This is due to network congestion resulting in packet-loss. For the controlled version where the bandwidth is kept at a maximum of 3 Mbps, the source and the receiver bandwidth are about the same because there is very little loss.

In our implementation with motion-JPEG, the most obvious difference in quality is the displayed frame rate at the receiver shown in figure 10. Note that when there is a high bit rate generated at the source, the amount of loss increases. In the uncontrolled version, the frame rate reduces to zero for a significantly long time. In contrast, the controlled version's frame rate reduces from 30 frames per second to an average of 15 frames per second and only occasionally drops below 10 fps.

6. Conclusion

The H.320 ITU Standard has defined a solution for videoconferencing over N-ISDN. This standard advocates a centralized approach to multiparty conferencing based on a MCU. Various aspects of the H.320 standard are clearly reusable in packet-switched environments while others are not. One aspect of the H.320 standard which is less clear as to its applicability to packet-switched network environments is the centralized approach to multiparty conferencing.

Packet-switched networks offer capabilities which may be exploited to offer multiparty videoconferencing solutions based on a distributed approach which offer advantages over the centralized scheme. We reviewed the various functions which an MCU can provide and then examined how these might be implemented in a distributed environment. In particular, we explored the multicast mechanisms which are currently being defined and applied them to a distributed multiway conference solution.

We described the Multimedia Multiparty Teleconferencing solution implemented in the Multimedia Networking group at the IBM T.J. Watson Research Center. This is an example of a distributed videoconferencing approach in a packet-switched environment. Numerous issues including rate control, multiple video stream composition, audio mixing and synchronization issues are described along with the chosen solutions for MMT. In general, a feasible distributed approach is described for the packet-switched environment, with demonstrable advantages over a centralized scheme.

Future video services will be supported at a variety of bit rates tailored to individual services, ranging from as low as 4 kbits/s in mobile services to as high as 135 Mbits/s for very high quality HDTV. These environments are increasingly important given the proliferation of high-speed, highly-reliable fiber networks and low-speed, lossy, wireless networks. In this context, it is crucial to investigate scaleable video delivery mechanisms, suitable for both stored video playout and real-time video conferencing, that would allow seamless connectivity between stations on this heterogeneous internetwork.

Compression gain is not the only criterion for selecting a video coder. Ease of manipulating in diverse communication environments will be increasingly important. It is here that techniques, such as those in which visual information is arranged in multiple layers of scalability, will provide the advantage over the largely inflexible traditional approach. Layered representations are very network friendly in that they make it possible to adapt video retrieval to the constraints of the access link (bandwidth limits of copper loops, radio links, etc.), and to extend available information to end users with a wide range of access capabilities, display capabilities, and computing environments.

Videoconferencing solutions are currently evolving from several directions. On the one side, there are the circuit-switched (e.g., Narrowband ISDN or the Switched-56Kbps phone lines) types of solutions, which being motivated by the telephony industry, can be likened to it. On the other side are the packet-based network (e.g., Ethernet and Token Ring legacy LANs) solutions, which endeavor to carry real-time traffic

over existing computer communications networks. The advent of ATM (Asynchronous Transfer Mode) [29] which promises to provide isochronous virtual connections over a cell-based network might eventually allow these two approaches to converge.

9. References

[1] "Narrow-band visual telephone systems and terminal equipment" ITU-T Recommendation H.320, July 1990.

[2] "Video CODEC for audiovisual services at p x 64 Kbit/s" ITU-T Recommendation H.261, July 1990.

[3] "Frame structure for a 64 to 1920 kbit/s channel in audiovisual teleservices" ITU-T Recommendation H.221, July 1990.

[4] "The system for establishing communication between audiovisual terminals using digital channels up to 2 Mbits/s" ITU-T Recommendation H.242, March 1993.

[5] "Multipoint control units for audiovisual systems using digital channels up to 2 Mbit/s" ITU-T Recommendation H.231, May 1992.

[6] "Generic Application Template" Draft Recommendation T.GAT, Feb. 1995.

[7] "Frame-Synchronous control and indication signals for audiovisual systems" ITU-T Recommendation H.230, March 1993.

[8] M. Willebeek-LeMair, Z. Shae, "Centralized versus distributed schemes for videoconferencing" Proceedings of the fifth IEEE Computer Society Workshop on Future Trends of Distributed computing Systems, pp.85-93, Korea, Aug. 1995.

[9] M-S. Chen. Z-Y Shae, D.D. Kandlur, T. Barzilai, and H. M. Vin. "A multimedia desktop collaboration system", Proceedings GLOBECOM 92, Dec. 1992.

[10] D. D. Kandlur, M. Willebeek-LeMair, T. Barzilai, Z-Y. Shae, and J-T Liu "Supporting timing requirements of digital video and audio in desktop conferencing systems" IFIP 6th international conference on high performance network, Spain, Sep. 1995.

[11] Z-Y Shae, P. Chang, and M. Chen "Capture and playback synchronization in video conferencing" SPIE proceedings on networking and communication, San Jose, Feb. 1995.

[12] D. Saha, D. Kandlur, T. Barzilai, Z-Y. Shae, and M. Willebeek-LeMair "A videoconferencing testbed in ATM: design, implementation, and optimizations" Proceedings of ICMCS 1995.

[13] M. Willebeek-LeMair, D. Kandlur, and Z-Y. Shae "On multipoint control unit for videoconferencing" proceedings of LCN, Oct. 1994.

[14] Z-Y. Shae and M-S. Chen "Mixing and playback of JPEG compressed packet videos" Proceedings of GLOBECOM'92 pp. 245-249, Florida, Dec. 1992.

[15] Y-C Chang, Z-Y Shae, and M. Willebeek-LeMair "Multiparty videoconferencing using IP multicast" SPIE proceedings on networking and communication, San Jose, Feb. 1996.

[16] M. Chen and Z-Y. Shae "Video Mixing technique using JPEG compressed data" United States Patent number 5257113, Oct. 26, 1993.

[17] D. Waitzman, etal "Distance Vector Multicast Routing Protocol" RFC 1075, Nov. 1988.

[18] A. Thyagarajan and S. Deering "Hierarchical Distance-Vector Multicast Routing for the MBONE", SIGCOMM'95.

[19] L. Wei, etal "The tradeoffs of multicast trees and algorithms" Internet Draft, Mar. 1995.

[20] J. Moy "Multicast Extensions to OSPF" RFC 1584, Mar. 1994.

[21] J. Moy "OSPF version 2" RFC 1583, Mar. 1994.

[22] A. J. Ballardie, etal "Core Based Trees: an architecture for scaleable interdomain multicast routing" SIGCOMM'93, Sep. 1993.

[23] A. J. Ballardie, etal "Core Based Tree Multicast - Protocol Specification" Internet Draft, June 1995.

[24] A. J. Ballardie "A new approach to Multicast Communication in a Datagram Internet work" Ph.D thesis, University College London, University of London, May 1995.
[25] “User-Network interface specification, version 3.1” ATM Forum, July 1994.
[26] S. Deering, etal "Protocol Independent Multicast: Motivation and Architecture" Internet Draft, Jan. 1995.
[27] S. Deering, etal "Protocol Independent Multicast: Protocol Specification" Internet Draft, Jan. 1995.
[28] F-C. Liaw “A Straw Man Proposal for ATM GROUP Multicast Routing and Signaling Protocol” ATM Forum/94-0995, Nov. 1994.
[29] “RTP: A transport protocol for real-time application” Internet Draft, draft-ietf-avt-rtp-05, July 1994.
[30] “Experimental Internet Stream Protocol, Version 2 (ST-II)” CIP working group, Oct. 1990.
[31] “ISO/IEC 13818-1 MPEG-II draft international standard system layer” March 1994.
[32] Cidon and I. S. Gopal, "PARIS: An Approach to Integrated High-Speed Private Networks," Intl. Journal of Digital and Analog Cabled Systems, vol. 1, pp. 77-85, 1988.
[33] L. Li, A. Karmouth, and N. Georganas, “Synchronization in real time multimedia data delivery” ICC’92, pp.587-591.
[34] Kanakia et al., “An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport”, Sigcomm 93.
[35] Bolot, Turletti, “A Rate Control Mechanism for Packet Video in the Internet”, Infocomm 94.
[36] Chakrabarti, Wang, “Adaptive Control for Packet Video”, IEEE Multimedia 94.
[37] Sakatani, “Congestion Avoidance for Video over IP Network” International COST 237 Workshop Proceedings.
[38] Pancha, P. and El Zarki, M., “MPEG Coding for Variable Bit Rtae Transmission” IEEE Communications Magazine, Vol.32 No 5, May 1994, pp. 54-66.
[39] J. Y. L. Boudec, “The asynchronous transfer mode: a tutorial” Computer Networks and ISDN Systems, vol. 24, pp.279-309, May 1992.