Design and Implementation of Multimedia Conference System on Broadcast Networks*

Jau-Hsiung Huang, Chun-Chuan Yang, Wei-Hsin Tseng,
Chung-Wei Lee, Biau-Jwo Tsaur, Lien-Kun Chuang, and Wen-Jen Liu

Communications and Multimedia Lab.
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan, ROC

Abstract

In this paper, an implementation of a multimedia conference system with shared white board developed on workstations under UNIX environment is presented. During the implementation, some design issues are encountered such as how to choose transport protocols, how to resolve problems of delay jitter, how to setup and manage the connections, how to resolve the lip synchronization problems, how to assign priority to packets and how to choose encoding schemes for audio and video. These design issues will be discussed in this paper.

1. Introduction

With the increasing power of modern computers and high speed network technologies, it brings the integration of multiple media into a single network application [1-6] possible. For people at different locations to communicate with one another, it is natural to provide the capability of video conference and shared white board.

Previously, we have implemented a video phone system on personal computers interconnected by token ring networks [6]. In that implementation, the limitation of the hardware (such as the processing power and bus bandwidth) and software (such as single-tasking operating system) prohibited the realization of some more powerful and friendly functions. However, with the proliferation of fiber optical local area networks and inexpensive workstations, a multimedia conference system based on workstations and FDDI networks becomes an attractive and feasible solution. Moreover, such a system can be used as a powerful tool for computer supported cooperative work (CSCW) environment.

The display of the system on the monitor is shown in Figure 1 which contains four video windows each with a resolution 320*240 and one graphic window for the shared white board. Among the video windows, three of those are used to show the video outputs of three of the remote speakers and the remaining one is used to show the video output of the local speaker. The window for local speaker is needed for the speaker to adjust his/her appearance on the monitor screen. The frame rate of the video is around 12 frames per second. Note that since data volume of video traffic is huge, a data compression board for video data is inevitable.

Figure 1. The display of the system on the monitor.

For the shared white board, the user can use the keyboard to type or use the mouse to draw pictures to be shared by all attendants in the conference. Note that various colors and functions are provided by the shared white board to users for drawing their pictures.

* This research was supported in part by National Science Council under grants NSC78-0414-E-002-325 and NSC78-0414-E-002-328.

There are many design issues to be taken into account in implementing such a system as follows. 1) How to establish a connection and how are they managed? 2) Whether a reliable (TCP) or unreliable (UDP) protocol should be used? 3) How to synchronize various media and how to resolve delay jitter problems created by random network delay? 4) How to assign priority to packets? 5) How to choose encoding schemes for audio and video traffic? These issues will be addressed in section three.

2. System Architecture and Software Environment

In this section, we first describe the hardware and software components of the system. The hardware of the system contains a group of SUN Sparc II workstations connected by an FDDI network. Other hardware requirements of the system follows. First of all, since the video application of the system generates a huge amount of video data to be carried by the network in real-time; hence, a video compression board is required in order to swiftly compress the video data. Normally, the compression ratio for video phone applications ranges between 20 and 200, depending on the picture resolution, encoding technology and video quality required. Currently, there are many compression standards for image and video applications such as JPEG [7], MPEG [8], and CCITT H.261 [9]. Among those, JPEG is an intra-frame coded algorithm while MPEG and H.261 are inter-frame coded algorithm.

For video processing, a video board is used in the system which compresses the video data using JPEG standard. JPEG standard is chosen for two reasons. First, it is an intra-frame video encoding algorithm such that error propagation will not occur. Second, it is commercially available at a lower cost compared to MPEG and H.261. This board also provides functions of A/D (Analog/Digital) and D/A conversion of video, i.e., the *frame grabber* capability. The compression ratio of JPEG in the application is around 25, which is acceptable for our system. For audio processing, the workstation has a built in function which supports μ-law PCM audio encoding scheme. Moreover, a hidden microphone and a speaker are also built in the workstation as the audio I/O devices. Lastly, one camcorder is used as the video input device.

The software environment is based on UNIX operating system, using OPENLOOK (OPENWIN) as the graphical user interface (GUI). The application program is coded in C language.

3. Design Considerations

Many design issues have to be taken into account for implementing the system. The topics include delay jitter, priority assignment of packets, video frame rate, network protocol, and coding techniques. We will discuss each of these issues in the following.

3.1 Choice of Transport Protocols

Before describing the network protocols used in the system, we should first discuss the characteristics of multimedia traffic. In [10], it was pointed out that 100% error free transmission of voice packets and video packets are not necessary as long as they are properly encoded. For example, since each video frame will stay on the screen for only 1/12 second and will then be covered by the following video frame; hence, by losing one video frame will only damage the display quality for 1/12 second, which is not very noticeable for human eyes. For voice packets, It has been shown in many experiments that digitized voice packets can suffer a packet loss percentage up to 2% without hurting the quality [11,12]. Based on these observations, we conclude that an error-free transmission of video and voice data is not necessary.

The network protocols used in the system is the TCP/IP protocol suite embedded in the UNIX operating system. Hence, the candidates for transport protocols are TCP, a protocol supporting reliable transmission, and UDP, a protocol supporting unreliable transmission. In [13], we showed that of the total processing time of TCP protocol in file transfering, more than 70% of the time is spent in data checksum computation. That is, data checksum is the processing bottleneck of TCP. Since there is no data checksum in UDP; hence, if the transmission error rate is small and an error free transmission is not required, UDP may be a better candidate than TCP for this application.

Moreover, since TCP requires that all packets should be delivered in sequence and error-free; therefore, any unreceived packet, may be due to transmission error or buffer overflow, will be retransmitted. Retransmission of packets incurs two side effects: 1) consuming extra bandwidth and 2) incurring extra delay. Since multimedia traffic is very delay-sensitive in one hand and may tolerate some packet loss in another, retransmission of packets may not be desired. Hence, UDP is a better candidate than TCP.

Last but never least, since TCP is an end-to-end transport protocol, it is impossible to support multicasting functions. On the other hand, since UDP is a connectionless protocol, therefore it presents no problem in implementing multicasting capability. Since the capability of multicasting is very important in implementing conferencing system; hence, UDP is again a better candidate than TCP. Based on the reasons stated above, UDP is chosen as the transport protocol for the system.

3.2 Connection Establishment and Management

In this subsection we will explain how to set up and manage the conference. First of all, a designated port number, say X, should be assigned to this video conference application in UDP domain. Then, the person who initiates the conference will connect to each of the intended attendants through port X for connection setup. During this setup process, a new port number, say Y, should be negotiated by all attendants to be used as the port for this conference such that port X can be released after the connection setup for other conferences. This feature allows a person to participate in more than one conference. After port Y is agreed by all attendants, every attendant will send his packets to port Y of all others to achieve multicast function.

Another benefit of this scheme is that since the negotiated port Y is known by attendants of this conference only; hence other people cannot intercept the packets of this conference. A better security is provided this way.

For the management of the conference, a conference chair, normally the conference initiator, is required during the conference. If the chair has to lease the conference early, a new chair has to be elected. The duties of the chair include the following. 1) To assign whose video output should be displayed on the video windows since there may be more than four people attending the conference while there are only four video windows. 2) To determine whether a new comer can join the conference. 3) To determine when to finish the conference. For the management of the conference, [5] provided a good reference covering many related issues.

3.3 Synchronization between Media and Delay Jitter of Packets

For video conference application, it is important to synchronize the voice and video data during the conversation. Since the video frame rate equals 12, that is, we have to send the video data of one frame every 1/12 second. During 1/12 second, there will be 667 bytes of voice data generated if the sampling rate equals 8 KHz. Hence, after sending the video data of one frame, the system will transmit all voice data corresponding to that video frame in order to synchronize voice and video data. However, since the data volume of one video frame is different from that of another after JPEG encoding; hence, the transmission time of the video data of one video is different from that of another. This situation creates delay jitter problem for voice traffic.

There are two other kinds of jitters for audio packets. One is resulted from the non-deterministic network access time for packet switched network and the other one is due to the operating system, in this case is UNIX, which does not support the real-time or isochronous scheduling of processes. For examples, in order to record audio data from the audio input device periodically to get the fixed length audio packets, an application needs the operating system to provide the fixed-time scheduling. Unfortunately, UNIX does not have such functions. Since voice traffic is very sensitive to delay jitter, hence, the quality of voice will be badly damaged if the delay jitter is serious. Therefore, to resolve problems of delay jitter is very important.

Traditionally, buffers are used to smooth the delay jitter of packet networks. Using buffers means introducing more delay of packets and it may produce unacceptable round-trip response delay. The experiences of our implementation show that the delay variance of network access can be ignored in FDDI networks. Therefore, our system simply plays/displays the audio/video packets when they are read from the socket without buffering in order to reduce delay.

To deal with the variant length of the audio packets, we use the non-blocking I/O functions of UNIX for reading data from the audio input device within some fixed interval which is specified in the program. Non-blocking reading from the audio device has the characteristic that the size of data read can be adjusted to some fixed values, i.e. 1024 * n bytes, where n = 0, 1, 2, .. , 8. It means that as long as we read the audio device fast enough, we can get either no audio data or a fixed length of 1024 bytes audio data that meets our need. But this method will not work properly when the process load of the operating system is too heavy to let us get the audio data within needed time interval. Therefore, the user should not put many background jobs when video conference is in use.

There is another method to deal with the problem of jitters which will record the amount of accumulated jitter. If the accumulated jitter exceeds a specified threshold, then all accumulated jitter will be removed. This method is more complex for we have to record the inter-arrival time and the size of the coming audio packet to compute the jitter caused by this packet. The threshold value to limit the accumulated jitter is determined by heuristic. We use an example shown in Figure 2 to illustrate this idea.

The sequence of audio packets shown in Figure 2 is A, B, C, D, and E and their respective sizes are 100, 50, 100, 50, and 50 ms. Note that the playing time of a packet can be calculated from its packet size. For instance, a packet of 1000 bytes requires a playing time of 1/8 second if the sampling rate equals 8K Hz. The interarrival time between two packets is shown in the row "interarrival time" in the figure, and their values are 30 ms (A~B), 80ms (B~C), 40ms (C~D),and 40ms (D~E). Since we adopt a no buffering policy, the arriving packet will be put into the buffer of the audio output device directly. Hence, we can use the size of packets and the interarrival time between packets to compute the jitter resulted from the arrived packet. For example, when packet A arrives, it needs 100ms to finish playing. Since packet B arrives only 30ms after packet A, a jitter of 100-30=70ms will be introduced, as shown in the row "jitter". Similarly, we can calculate

other jitters resulted from C (50-80=-30), D (100-40=60), and E (50-40=10). In this way, we can calculate the accumulated jitter up to the time when a audio packet arrives, as shown in the row "Σjitter". When we detect the Σjitter exceeds the threshold value (100 ms in this case), we remove all accumulated jitter by clearing the device buffer and set Σjitter to zero. By so doing, the delay jitter will not accumulate and we do not have to clear the device buffer after every packet arrival.

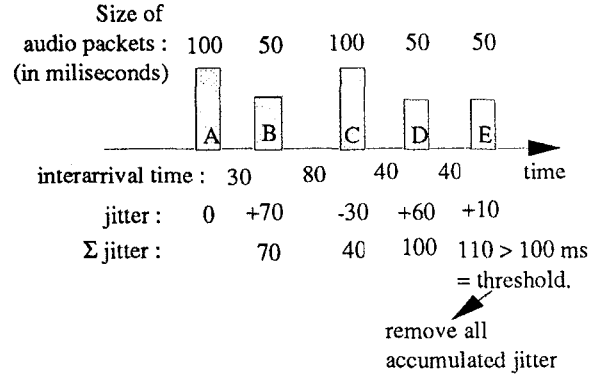

Figure 2. An example of audio packet arrival pattern and jitters.

3.4. Priority assignment

Since voice is more delay sensitive than video and other media, hence voice should be assigned the highest priority with video second. Clearly, the real-time hand-drawn images have the third highest priority and data file transfer should be assigned the lowest priority. The capability of supporting priority is the reason why FDDI network is chosen as the network platform. Such a design allows other people to transfer files without hurting the quality of voice or video. Hence, this network can still support other general purpose applications while someone is running the video conferencing application.

3.5. Choice of encoding schemes

Since UDP is used as the transport protocol as explained earlier, it is possible that video packets or voice packets may be lost without retransmission. This phenomenon has an impact on choosing the techniques of video coding and voice coding. We first examine the coding technique for voice. The most popular voice coding techniques may be PCM, DPCM, and ADPCM. Since both DPCM and ADPCM encode the difference between voice samples to gain a higher compression ratio, therefore a correlation between voice packets exists. The disadvantage of such schemes is that by losing one voice packet will affect all following voice data (called error propagation). Hence, the voice coding technique employed in the system is simply PCM.

Similarly, among three standard video coding techniques, JPEG is an intra-frame encoding scheme which encodes the video data frame by frame without any correlation between frames. On the other hand, MPEG and H.261 are inter-frame encoding schemes which normally have a higher compression ratio but will create correlations between frames. Hence, MPEG and H.261 also have the problem of error propagation. Therefore, JPEG encoding technique is employed in the system to avoid error propagation although the compression ratio is slightly lower than that of the other two schemes.

4. Discussions and Conclusions

We will calculate the number of participants allowed to use the video conference concurrently. Since the resolution of the video window is 320*240 with 24 bits for each pixel for color display; hence the data rate of the video before JPEG compression is 320*240*24*12 = 18.4 Mbps for 12 frames per second. After the compression, the data rate roughly equals 800Kbps; hence the compression ratio is about 23. The voice data encoded by PCM requires a network bandwidth of 64 Kbps; hence, the total network bandwidth required by one person is 864 Kbps. If we use half of the network bandwidth, i.e., 50Mbps, for video and voice transmission, then the system can support about 60 users concurrently. Clearly, more users can be supported if a lower resolution of windows is used.

With the increasing processing power of workstations and multitasking capability of the operating systems, users can actually enjoy more than video conference and shared white board. For example, the user can simultaneously open another window running spread sheet. Further, the user can extract data from the spread sheet to discuss with others in the conference. Hence, such a system is actually a powerful tool for computer supported cooperative work (CSCW). Lastly, although the system is implemented on an FDDI network, such a system can easily be ported to an ISDN network if the network is available.

References

[1] J. B. Postel, G. G. Finn, A. R. Katz, and J. K. Reynolds, "An experimental multimedia mail system," ACM Trans. Off. Inform. Syst. vol. 6, no. 1, pp. 63-81, Jan. 1988.

[2] C. Nicolaou, "An architecture for real-time multimedia communication systems," IEEE J. Select Areas Commun. vol. 8, no. 3, pp. 391-400, Apr. 1990.

[3] P. V. Rangan, and D. C. Swinehart, "Software architecture for integration of video services in the etherphone system," IEEE J. Select Areas Commun. vol. 9, no. 9, pp. 1395-1404, Dec. 1991.

[4] W. H. F. Leung, T. J. Baumgartner, Y. H. Hwang, M. J. Morgan, and S. C. Tu, "A software architecture for

workstations supporting multimedia conferencing in packet switching networks," IEEE J. Select Areas Commun. vol. 8, no. 3, pp. 380-390, Apr. 1990.

[5] M. Arango et. al., "The Touring Machine System," Communications of the ACM, Vol.36, No.1, pp.68-77, January 1993.

[6] Jau-Hsiung Huang, Wei-Hsin Tseng, Ming-Jeu Ding, Yo-Song Su and Lin-Chih Wu, "VirtualTalker: An On-Line Multimedia Systems on Token Passing Networks," to appear in Proc. International Conference on Consumer Electronics, Illinois, June 1993.

[7] G.K. Wallace, "The JPEG Still Picture Compression Standard," *Communications of the ACM*, Vol.34, No.4, pp.30-44, Apr. 1991.

[8] *MPEG standard draft ISO-IEC/JTC1 SC29 on 22*, November 1991.

[9] 1989 CCITT Study Group XV, TD 35. Draft revision of recommendation H.261: Video codec for audiovisual services at px64 kbits/s. *Image Communication*, pp.221-239, August 1990.

[10] Jau-Hsiung Huang and Shen-Horng Lee, "MHTP: A Multimedia High-Speed Transport Protocol," Proc. IEEE Globecom'92, December 1992, pp.1364-1368.

[11] T. M. Chen and D. G. Messerchmitt, "Integrated voice/data switching," *IEEE Communications*, Vol.26, pp.16-26, June 1988.

[12] N. S. Jayant and S.W. Christense, "Effects of Packet Losses in Waveform Coded Speech and Improvements Due to an Odd-Even Sample-Interpolation Procedure," *IEEE Trans. Communication*, Vol.COM-29, pp.101-109, February 1981.

[13] Jau-Hsiung Huang and Chi-Wen Chen, "On Performance Measurements of TCP/IP and its Device Driver," Proc. 17th Annual Local Computer Network Conference, Miniapolis, Sept. 1992, pp.568-575.