Copyright Notice

Institute of Communication Networks and Computer Engineering
University of Stuttgart
Pfaffenwaldring 47, D-70569 Stuttgart, Germany
Phone: ++49-711-685-68026, Fax: ++49-711-685-67983
Email: mail@ikr.uni-stuttgart.de, http://www.ikr.uni-stuttgart.de

Controlling QoS in a Collaborative Multimedia Environment

Marco Alfano and Rolf Sigle
International Computer Science Institute
1947 Center Street
Berkeley, CA 94707
{alfano,sigle}@icsi.berkeley.edu

Abstract

A collaborative multimedia environment allows users to work remotely on common projects by sharing applications (e.g., CAD tools, text editors, white boards) and simultaneously communicate audiovisually. Several dedicated applications (e.g., MBone tools) exist for transmitting video, audio and data between users. Due to the fact that they have been developed for the Internet which does not provide any Quality of Service (QoS) guarantee, these applications do not or only partially support specification of QoS requirements by the user. In addition, they all come with different user interfaces.

In this paper, we first discuss the problems that we experienced both at the host and network levels when executing a multimedia application and varying its resource requirements. We then present the architectural details of a collaborative multimedia environment (CME) that we have been developing in order to help a user to set up and control a collaborative multimedia session.

1. Introduction

The increasing availability of broadband networks allows the deployment of new services for an ever growing number of possible users. Among these new services, a great deal of interest has been addressed towards real-time and interactive applications, e.g., videoconferences and shared document editors, particularly because of the worldwide and decentralized features of today's research and development organizations.

A collaborative multimedia environment allows users to work remotely on common projects by sharing applications (e.g., CAD tools, text editors, white boards) and simultaneously communicate audiovisually. In order for a collaborative multimedia environment to be widely used, it should utilize the same system resources (hosts and networks) that users normally have available (e.g., PCs, workstations, Internet). However, this requires that the same environment has to be shared by multimedia applications with strict requirements (e.g., real-time) and other applications that do not have comparably strict performance requirements.

Presently, there is no globally available mechanism for managing system resources that discriminates among applications privileging, for example, the real-time ones. Moreover, different policies are used to manage different resources and the management of the different resources is often not coordinated, particularly when the resources are distributed.

Several dedicated applications (e.g., MBone tools [1]) exist for transmitting video, audio and data between users. Due to the fact that they have been developed for the Internet which does not provide any Quality of Service (QoS) guarantees, these applications do not or only partially support specification of QoS requirements by the user. In addition, they all come with different user interfaces.

In this paper, we present the architectural details of a collaborative multimedia environment (CME) that we have been developing in order to help a user to set up and control a collaborative multimedia session. The paper is organized as follows: Section 2 presents some experimental measurements on the execution of multimedia applications that led us to the idea of building a CME with QoS control. Section 3 presents the architectural details of the CME. Sections 4 and 5 respectively describe the QoS mapping and the QoS control mechanism. Finally, Section 6 presents some conclusions and a discussion on the future work.

2. Experimental measurements on video provision

We ran some tests in order to analyze how host and network resources influence the execution of multimedia applications. For our tests, we chose only one media (video) unidirectional communication because, by doing so, we could examine on the receiving host just the CPU consumption for decoding video frames.

As a sending host, we used a Sun™ sparc20 workstation equipped with a Parallax™ video board. This video board supports JPEG compression in hardware. Thus, we could vary the video frame rate in a wide range without consuming a lot of CPU resources. As a receiving host, we used a Sun™ sparc5 workstation with a Sun Video™ board. This video adapter does not provide the same hardware support as the Parallax™ board, therefore the decompression has to be done in software and this requires a lot of CPU resources.

For these tests we considered an environment for world-wide collaboration with limited but guaranteed network resources. We used the MAY (Multimedia Applications on Intercontinental Highway) network, an ATM network connecting North America to Europe [2]. We set up a loopback in Germany thus obtaining a world-wide ATM link with a fixed transmission rate of 1.5 Mb/s.

We used the Mbone tool vic [3] to transmit video. While changing the sending video frame rate, we observed the displayed video frame rate, the CPU consumption on the receiving host and the transmission rate of the ATM network. The video frame rate was provided by the vic itself. To measure the CPU consumption we used the UNIX "top" utility. Although top does not give very precise results, we could still see the general trend. We measured the transmission rate of the ATM network using the management tools of our Synoptics™ ATM switch.

By sending video with an increasing frame rate, we obtained the results shown in Figure 1. The three graphics show the influence of both CPU and network resources on the displayed frame rate. At a sending rate of 16 fps, the CPU got saturated and at 19 fps also the network became congested leading to the dramatic reduction of the displayed frame rate.

In [4] we present the results of a second set of experiments that we executed by using our local ATM network (155 Mb/s). Although we did not experience network congestion, we still observed the saturation of the receiving CPU when increasing the sending frame rate. This caused random discard of video frames on the receiving host and consequently a subjective decrease of the video quality.

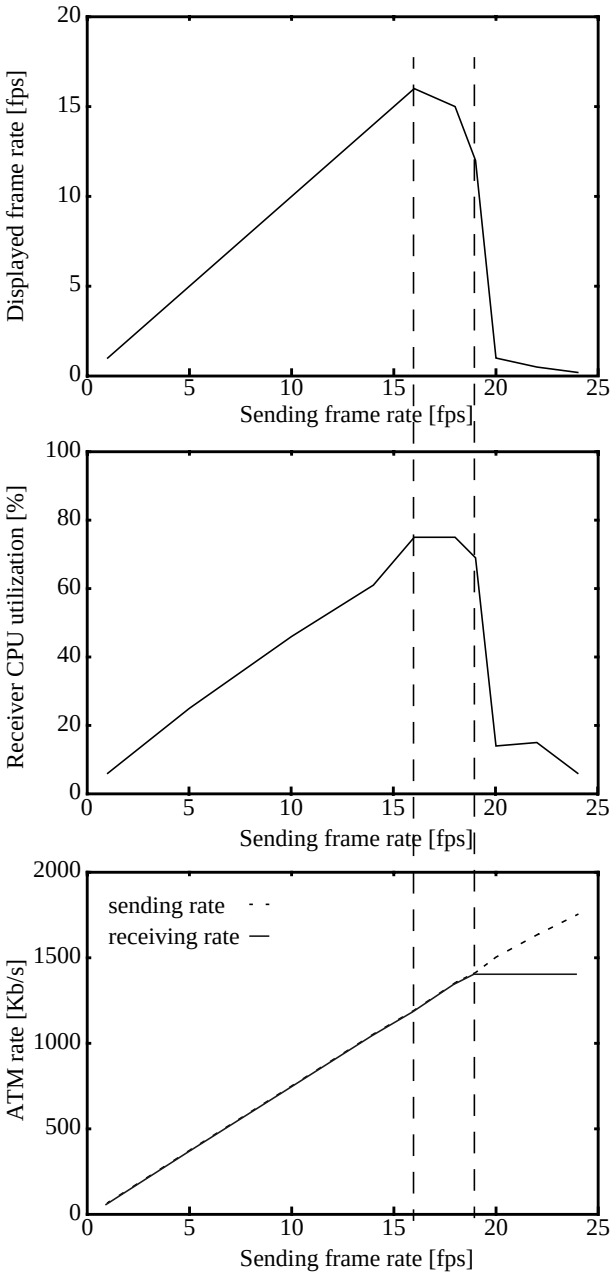

Figure 1. Experimental results with host and network congestion.

3. The collaborative multimedia environment (CME) architecture

Our experiments, as discussed above, show that applications often suffer quality degradation during a multimedia session even though transmission capacity is sufficiently high. Quality decrease is often caused by resource saturation at the receiving host.

Since host resources play a crucial role in executing multimedia applications, they should be taken into account, beside network resources, when executing multimedia applications with QoS guarantees. Some work on QoS guarantees has been carried out separately at the network level [5], [6] and at the host level [7].

We have been developing a collaborative multimedia environment (CME) that realizes an efficient use of both host and network resources while providing the user with a facility to easily start a collaborative session and control the QoS parameters of each media. Our CME consists of

integrated multimedia applications, one for each user (Figure 2).

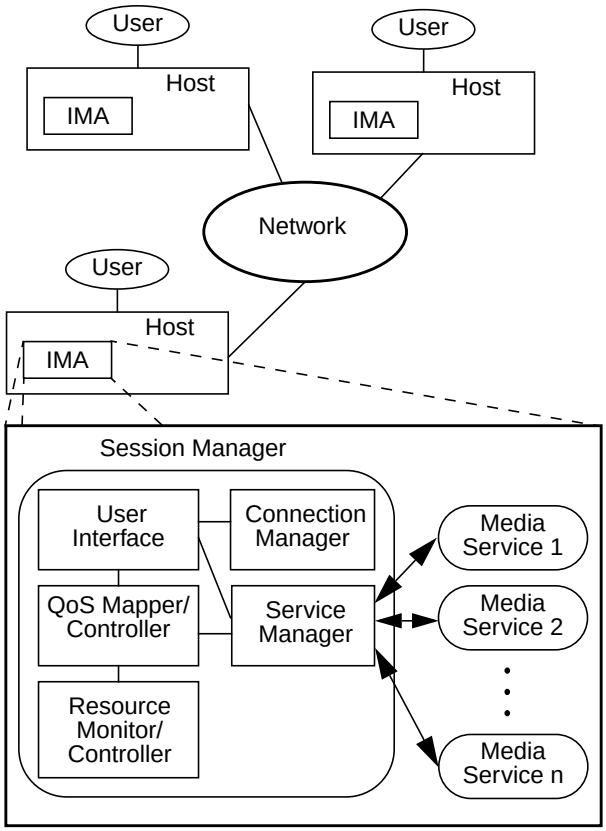

Figure 2. Collaborative multimedia environment.

The integrated multimedia applications contain different *media services* (i.e., video, audio and data services) which are controlled by a *session manager.* Each media service provides basic functions (e.g., sending, receiving and displaying video frames) and interacts with devices (or servers controlling devices) in its media category. While media-specific details are relegated to the media services, common functions are assigned to the session manager. It provides general mechanisms for session-related tasks (invite, join, disconnect, start media services, QoS control,...). The session manager is made up of the following components:

- A *connection manager* for establishment and disconnection of collaborative sessions. During session establishment, other users are invited to join the session. Since any connection manager can initiate a collaborative session, the collaborative session does not rely on any centralized session moderator but is based on a distributed peer-to-peer model.
- A *QoS mapper/controller* that translates user QoS requirements into parameters for the media services and into QoS requirements for the underlying resources (i.e., host and network resources). It also executes a control mechanism in order to satisfy the user requirements on the media services.
- A *resource monitor/controller* of the host and network resources used for the different media services.
- A *service manager* for starting and stopping user-requested media services for the session. The service manager also monitors and changes the service parameters (e.g., video frame rate) following the indication of the QoS mapper/controller.
- A *user interface* that provides a graphical interface for starting or joining a collaborative session. Through this interface, a user can specify the media services he wants to use in the session and change his QoS requirements on the services.

A user who wants to start a collaborative session, specifies through the graphical interface the users he wants to invite to the session and the media services to be used. The connection manager contacts the invited users who will receive a message containing the name of the inviting person and the media services he wants to use. An invited user can either accept or refuse to join the session. Besides, he can specify that he will join the conference with a subset of the proposed media services because, for example, one media service may not be available due to the lack of hardware support. When this setup phase has been completed, the service managers at the different hosts will start the provision of the chosen media services with some default values and the collaborative session will take place.

During the collaborative session, a user can change his QoS requirements on the media services. QoS requirements at user level are specified by means of simple attributes (e.g., low, medium, and high quality video). These “high-level” attributes are translated by the QoS mapper/controller into parameters for the media services and into QoS requirements for the underlying resources. Thus, depending on the user requests for the media services, the QoS mapper/controller decides the performance parameters for the services (e.g., receiving video at 10 fps) and evaluates through the resource monitor/controller whether these parameters can be supported by the underlying resources. Finally, the QoS mapper/controller will make the necessary adjustments so that the media services can perform as planned.

In the next two sections, we describe the QoS mapping and the Qos control mechanism into details.

4. QoS mapping

Since this architecture is oriented towards the end user, he must be able to express his QoS requirements for the media services in a simple way (e.g., low, medium, or high quality video). These requirements will then be translated on one hand into parameters for the media services (e.g., frame rate for video) and on the other hand in QoS requirements for the underlying resources. As said above, the architecture component that provides the QoS translation at the different levels is the QoS mapper/controller.

Let us now examine how QoS requirements can be expressed at different levels. At the user level, QoS requirements can be grouped into two categories:

- *direct requirements*, when the user explicitly specifies a requirement for a media service, e.g., he asks for high quality video;
- *indirect requirements*, when the user makes some actions, such as iconifying a video window, that indirectly makes suggestions about the user interest on a particular media service (in the case of iconifying a video window, the user shows no interest for that video).

While the user actions that lead to indirect requirements can be detected by examining the user environment and the way he interacts with it (e.g,. iconifying or deiconifying a window, putting a window in background), more effort is required to define direct requirements. A direct requirement should be at the same time easy to understand for the user, general enough to include more particular requirements (so that the user is not required to specify too many requirements), and tractable so that it can be translated in QoS requirements for the underlying resources.

A proper way to express user requirements entails a detailed analysis on how a user expects a media service to behave more or less properly and how the satisfaction of the user for the media service quality can be expressed in quantitative terms. In what follows, we present the first results of an analysis on user requirements that we have been carrying out.

In order for the user not to deal with too many parameters, we define only one global requirement for each media and we indicate it with the generic term of *quality.* Thus, we will have video quality, audio quality and so forth. The quality requirement is a repository of more specific requirements on a media service. For example, video quality is intended in a broader sense than just considering how good received video pictures are compared to the original ones. This is, of course, part of video quality and is related to spatial vision but there is also temporal vision that must be taken into account, i.e., how the user perceives scene changes in the received video compared to the original one [8].

Many studies in the literature dealing with quality estimation of digitally coded video sequences [8], [9] and audio sequences [10], [11] use a five level scale, reported in Table 1, for quality rating. This scale is also used for subjective testing in the engineering community [12].

Rating	Impairment	Quality
5	Imperceptible	Excellent
4	Perceptible, not annoying	Good
3	Slightly annoying	Fair
2	Annoying	Poor
1	Very annoying	Bad

Table 1. Quality rating on a 1 to 5 scale.

We use the same five-level scale to define the quality of a media service and we give a user the possibility to specify one of these levels as a way to express his requirements. In the case of video, this scale is used to assess quality for both spatial and temporal perception [8]. In practice, the user will use a slider for each media service to indicate his quality requirements from a minimum value (quality level 1) to a maximum value (quality level 5).

Once the way to express user requirements has been defined, the next problem is to find a mapping between user requirements (quality levels) and parameters of the media services. The question is, what is the performance a media service must have in order to provide a certain quality level?

We need some mapping functions that connect, for example, video quality to video frame rate. These functions are similar to the “benefit functions” found in [13] and require the execution of subjective tests in order to determine whether for a given performance the user perceives the quality level of the media service as bad, poor, fair, good, or excellent.

We executed some tests with a group of ten people and asked them to rate the quality of video and audio sequences according to the one-five scale discussed above. As a video application, we used vic [3] and as an audio application, we used vat [14]. In the first test, we showed a video sequence at different frame rates in order to evaluate the temporal quality of video. First, we showed the video sequence at 30 fps. We told the viewers to consider the quality of that video sequence to be five and to rate the quality of the next video sequences against that one. In the second test, we showed a video sequence with different resolutions in order to evaluate the spatial quality of video. First, we showed the video sequence with the best resolution the video application could provide. Again, we told the viewers to consider the quality of that video sequence to be

five and to rate the quality of the next video sequences against that one. We then varied the resolution as a percentage of the best provided resolution. As a final test, we transmitted an audio stream with different encoding schemes in order to evaluate the audio quality. The PCM encoding scheme was considered to provide quality five.

We averaged the obtained results and built the tables that map the quality levels to the service parameters. The results of this mapping are reported in Table 2 for video and in Table 3 for audio.

Quality	Frame rate (fps)	Resolution (%)
5	25 - 30	65 - 100
4	15 - 24	50 - 64
3	6 - 14	35 - 49
2	3 - 5	20 - 34
1	1 - 2	1 - 19

Table 2. Video quality rating for JPEG video.

Quality	Encoding Scheme
5	PCM
5	PCM2
5	PCM4
4	DVI
4	DVI2
4	DVI4
3	GSM
2	LPC4

Table 3. Audio quality rating.

The third and final step in QoS mapping is to translate the media service parameters in QoS requirements for the host and network resources. Different resource parameters can be connected with the performance of a media service. For simplicity, we consider QoS requirements for the following resource parameters:

- Network Resources {bandwidth (Kb/s)};
- Host Resources {CPU type, %CPU}.

For video, it is very difficult to correlate media service performance and requirements on resources. Network bandwidth and mainly CPU utilization are very influenced by the frame size (assuming the user has the possibility to change the video size), compression scheme, and degree of movement (slow or rapid scene changes).

Considering the mapping between the quality levels and the media service parameters discussed above, we estimated the resources that are needed to obtain the different quality levels. The results are reported in Table 4 for video and in Table 5 for audio. For video, we estimated the necessary resources for receiving JPEG video (320 x 240). For each quality level, we considered three possible degrees of movement, i.e., still, slow motion, and high motion. As a receiving host, we used a Sun™ sparc5.

Quality	Degree of movement	Frame rate (fps)	Resolution (%)	Bandwidth (Kb/s)	Used CPU (%)
5	High motion	25	65	1700	> 100
5	Slow motion	25	65	1650	> 100
5	Still	25	65	1600	49
4	High motion	15	50	840	> 100
4	Slow motion	15	50	820	69
4	Still	15	50	800	37
3	High motion	6	35	270	38
3	Slow motion	6	35	260	34
3	Still	6	35	260	21
2	High motion	3	20	102	16
2	Slow motion	3	20	102	14
2	Still	3	20	100	7
1	High motion	1	1	17	6
1	Slow motion	1	1	16	6
1	Still	1	1	16	5

Table 4. Mapping of video quality to resources for JPEG video.

Quality	Encoding Scheme	Bandwidth (Kb/s)	Used CPU (%)
5	PCM	68	< 1
5	PCM2	66	< 1
5	PCM4	64	< 1
4	DVI	38	~1
4	DVI2	35	~1
4	DVI4	34	~1
3	GSM	15	~26
2	LPC4	7	~11

Table 5. Mapping of audio quality to resources

5. The QoS control mechanism

The user who starts the collaborative session chooses the media services to be used for that session. The media services are started with some default values and each participant in the session is presented a graphical interface that contains information on the media services used in the session (Figure 3). For each media service there is a meter that goes from zero to five and indicates the quality of that service. Level zero indicates that the service is not being received.

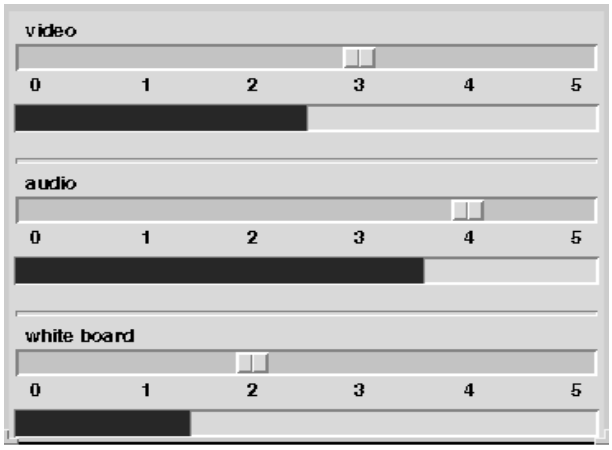

Figure 3. Control window for the media services.

The service manager will monitor the media services and report the values of the service parameters to the QoS mapper/controller which in turn will translate these values to quality levels and pass them to the graphical interface for displaying. If a quality level is connected with more than one service parameter, the QoS mapper/controller will use the following expression to compute a single value for the service quality (*Service_Q*):

$$Service\_Q = \alpha_1 Q_1 + \alpha_2 Q_2 + \ldots + \alpha_n Q_n$$

where:

- Q_i is the quality level obtained for parameter P_i;
- α_i is a weight indicating the relative importance of P_i for *Service_Q*; and
- $\alpha_1 + \alpha_2 + \ldots + \alpha_n = 1$,

For a video service, for example, we will have:

$$Video\_Q = \alpha_1\ frame\_rate + \alpha_2\ resolution$$

Normally, a user will not use all the media services involved in a collaborative session at the same time. For example, at the beginning of a collaborative session, a user usually wants to see and talk to the other participants in order to exchange the basic ideas on the common work and decide how to proceed with it. Once started working on a common text/picture, the interests of the users are mainly directed to the shared application. Users are not too interested in seeing each other any more, however, they want to keep talking to each other. Thus, user interests on the different media services are likely to change over time during a collaborative session.

The graphical interface provides an easy way to express user requirements by means of a slider that allows specification of the quality level for each media service. The slider moves in a discrete way and the user may specify any of the five levels that correspond to the five quality levels discussed above. In addition, specifying zero, the user indicates that he does not want to receive that service. An indirect requirement, as iconifying a video window, will have the same effect as to move the slider to zero.

The indication of the slider is two-fold:

1) it indicates the quality the user wishes to perceive for that media service;
2) it indicates the interest of the user for that service compared to the other services by assigning a priority to the service. The priority corresponds to the quality level with five being the highest priority and zero being the lowest.

The control mechanism, in trying to satisfy the user requirements, will establish a priority list of services based on the assigned priorities and will privilege more those services with higher priorities. The control mechanism will be activated when the quality level chosen by the user differs from the actual value supplied by the system beyond a threshold for a given time interval (to avoid having continuous control activity). This may happen because the user changes his requirements on a media service either directly through the slider or indirectly, e.g., iconifying a video window. Moreover, the status of the resources may not allow a media service to perform in a way that is even close to the quality level requested by the user.

The control mechanism will try either to change the media-service parameters or to reassign the resources so to satisfy the user requirements. To this end, it monitors the status of the host and network resources through the resource monitor/controller.

We can consider different scenarios depending on which extent the resource monitor/controller can control resources. If it is able to reserve both network and host resources, the QoS mapper/controller will recompute the service priorities and, based on these priorities, it will reas-

sign resources to services trying to guarantee the quality of the services with higher priorities. A user requirement then becomes an objective for the whole system that should properly act in order to satisfy this requirement [15]. Unfortunately, as we said above, presently there is no globally available mechanism for managing system resources that discriminates among applications. This is mainly due to the fact that a global network like the Internet does not provide any mechanism to manage it as a whole. Moreover, the Internet does not give the possibility to privilege some applications allocating a minimum amount of resources to them.

A more realistic scenario assumes that the session manager can partially control resources, i.e., it can control host resources but not network resources. Hosts usually belong either to users or to institutions where users work. This means that users (or system administrators who act on users' behalf) can directly access host resources. On the other hand, users and system administrators cannot control the resources of the networks through which the data of the collaborative session will flow, unless all the participants are in the same local network. Thus, the particular assumption that users can only control their host resources seems quite reasonable.

In this scenario, when the QoS mapper/controller observes a disequilibrium between the request of the user and the achieved performance of a media service, it will try to understand whether the system resources are not sufficient or whether the problem is caused by a wrong setting of the service parameters. Let us consider, for example, the case where the service meter indicates that a media service is performing with a lower quality level than the one requested by the user.

The control mechanism first checks whether the reason of poor performance of a media service is related to the underlying resources that, for example, get saturated. In this case, the control mechanism determines whether the problem is either in the network or in the host. If the problem is the host CPU, the control mechanism reassigns the CPU resources in order to increase the performance of that service while, at the same time, respecting the priorities of the services. If the problem lies in the network or the CPU is not powerful enough to support the service requirements, the QoS mapper/controller asks its peer on the sending site to lower the parameters of the media service. In fact, as shown in Figure 1, sometimes decreasing the values of the service parameters on the sending site may entail a better performance of the service on the receiving site and consequently a higher quality for the user.

Whenever the QoS monitor/controller of a sending site receives a request from its peer on a receiving site for either increasing or decreasing the value of the parameters of a media service, it will behave differently depending on whether the collaborative session is between two users or more. In the former case, the sending site will immediately act upon the request of the receiving site. In the latter case, the sender will not act immediately but will store the request. As soon as it receives coherent requests from a specified number of sites (e.g., the majority of sites), it will change the service parameters as requested.

If the QoS mapper/controller on the receiving host does not sense any problem in the network and in the CPU, it asks its peer on the sending side to increase the value of the parameters of that service so that it can perform as expected. On receiving a request from the receiving site, the QoS mapper/controller on the sending site will behave as described above

If the user requires either directly or indirectly quality zero for a media service, the QoS mapper/controller will ask the service manager to stop temporarily receiving that media service and will inform its peer on the sending site that it is not interested in receiving that service. As soon as the QoS mapper/controller of a sending site realizes that nobody is interested in receiving a service, it will stop temporarily transmitting the related media stream, thus avoiding a useless waste of resources.

6. Conclusion and future work

In this work, we first discussed the problems experienced both at the host and network levels when executing multimedia applications with varying resource requirements. We then presented the architectural details of a CME that we have developed in order to help the user to set up and control a collaborative multimedia session.

We are currently implementing a prototype of the CME architecture on Sun™ sparcstations. As media services, we use the MBone tools developed at the UC Berkeley and Lawrence Berkeley Laboratory, i.e., the *video conferencing tool (vic)* [3] for video, the *visual audio tool (vat)* [14] for audio and the *white board tool (wb)* [16] as a white board. For vic and vat, we give a user the possibility to start and stop the services and to change the QoS whereas, for wb, we presently only give a user the possibility to start and stop the service.

The resource monitor/controller is a process that continuously monitors the resources. We use the "top" UNIX utility to measure the CPU load and the real-time feature of the Solaris™ operating system to control the allocation of the CPU to the processes. Besides, we directly get information on bandwidth and packet losses from vic and vat by using the RTP protocol [17]. Although RTP provides us with information at the application level (global statistics on the combined effect of network and CPU), we can still understand whether a potential resource saturation is either

in the network or in the CPU because we directly monitor the CPU. The service manager is a process that monitors the parameters of the media services and passes them to the QoS mapper/controller for the translation into quality levels. The monitoring of the service parameters is done through the "send" Tcl/Tk function that interacts with vic and vat that have been partially built using the Tcl/Tk interpreted language [18]. The QoS mapper/controller has a controlling process that starts whenever there are decisions to make, checks the status of the resources and makes its choices as described in the previous section.

The work presented in this paper, to our best knowledge, is one of the first attempts in creating an integrated architecture for QoS control of a collaborative multimedia environment that spans from the user level down to the resource level. There are still different open issues that require further investigation. Among them, a better understanding of user requirements is necessary in order to evaluate whether the generic user is comfortable with the quality levels introduced here. Moreover, it is important to understand whether a user should have the possibility to express more than one requirement for a media service, e.g., for video, he could express his requirements for temporal quality (frame rate) and spatial quality (picture resolution) separately. More work is required in mapping user requirements into media-service parameters and system resources. Other service and resource parameters should be taken into account beside the ones already considered here and their influence on the media-service quality should be evaluated. More work also needs to be done for the control mechanism. In particular other scenarios should be considered beside the one that assumes that a user can control host resources but not network resources. We plan to investigate how to control the different resources in an integrated way in order to guarantee that a user obtains the service quality he is requesting for.

References

[1] V. Kumar. *MBone: Interactive Multimedia On The Internet*. Macmillan Publishing, Simon & Schuster, 1995.

[2] Multimedia Applications on Intercontinental Highway (MAY). On line description, *http://www.icsi.berkeley.edu/MAY/index.html*.

[3] S. McCanne and V. Jacobson. vic: A flexible framework for packet video. *Proc. of ACM Multimedia'95*, pp. 511-522, San Francisco, November 1995.

[4] M. Alfano, R. Sigle and R. Ulrich. Management of cooperative multimedia sessions with QoS requirements. *Proc. of IEEE Gigabit Networking Workshop GBN `96*, San Francisco, March 1996.

[5] D. Ferrari, A. Banerjea and H. Zhang. Network Support for Multimedia - A Discussion of the Tenet Approach. *Computer Networks and ISDN Systems*, vol. 26, pp. 1267-1280, July 1994.

[6] L. Zhang et al. RSVP: A new ReSerVation Protocol. *IEEE Network*, vol. 7, pp. 8-18, September 1993.

[7] D.P. Anderson. Metascheduling for continuous media. *ACM Transactions on Computer Systems*, vol. 11, pp. 226-252, August 1993.

[8] C.J. van den Branden Lambrecht and O. Verscheure. Perceptual quality measure using a spatio-temporal model of the human visual system. *Proc. SPIE Int.l Symp. on Visual Communications and Image Processing '96*, Orlando, March 1996.

[9] A. Basso et al. Study of MPEG-2 coding performance based on a perceptual quality metric. *Proc. PCS 96*, Melbourne, 1996.

[10] W.R. Daumer. Subjective evaluation of several efficient speech coders. *IEEE Trans. on Communications*, pp. 655-662, April 1982.

[11] W.C. Treurniet and L. Thibault. Perceval - A model for objective perceptual assessment of audio. On line publication, *http://www.crc.doc.ca:80/crc/branches/DRB/list.html*.

[12] ITU-R Recom. BT.500.7. Methodology for the subjective assessment of the quality of television pictures.

[13] L.C. Schreier and M.B. Davis. System-level resource management for network-based multimedia applications. *Proc. NOSSDAV'95*, Durham, April 1995.

[14] V. Jacobson and S. McCanne. vat - LBNL Audio Conferencing Tool. On line description, *http://www-nrg.ee.lbl.gov/vat/*.

[15] M. Alfano. Scheduling features in distributed systems. *Proc. of the SBT/IEEE International Telecommunications Symposium*, pp. 52-56, Rio de Janeiro, August 1994.

[16] V. Jacobson and S. McCanne. wb - LBNL Whiteboard Tool. On line description, *http://www-nrg.ee.lbl.gov/wb/*.

[17] H. Schulzrinne et al. RTP: A transport protocol for real-time applications. *IETF RFC 1889*, January 1996.

[18] J.K. Ousterhout. *Tcl and the Tk Toolkit*. Addison-Wesley, 1994.