2M11.2835.8

Université de Montréal

Quality of Services Adaptation Model for Distributed Multimedia Application Based on RTP protocol on IP network

par

Yong Guo

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté a la Faculté des études supérieures
En vue de l'obtention du grade de
Maître en Informatique

July, 2000

Faculté des Études Supérieures
Grade octroyé
à compter du
11 JAN. 2001
Université de Montréal

© Yong Guo, 2000

QA
76
U54
2001
v.004

Université de Montréal
Faculté des études supérieures

Ce mémoire de maîtrise intitulé

Quality of Services Adaptation Model for Distributed Multimedia Application Based on RTP protocol on IP network

Présenté par

Yong Guo

A été évalué par un jury composé des personnes suivantes :

Présidente:	Esma Aïmeur
Directrice de recherche:	Rachida Dssouli
Membre:	Brigitte Kerhervé

Mémoire accepté le : 16-11-2000

Acknowledgments

I would like to thank Prof. Rachida Dssouli gives me many valuable suggestions and great helps during the whole research process. It is a great pleasure to work with our team member Mathieu Poirier and Jean-Marc Ng, thanks for their discussions and suggestions. I will never forget the “struggles” we made for our project. Great thanks are also given to Prof. Jian-Yun Nie and Mohamed Salem.

Résumé

Le déploiement massif de l'internet s'accompagne d'un vaste développement des applications multimédias distribuées. Les domaines d'utilisation de ces applications sont très nombreux : télé-enseignement, systèmes de diffusion à la demande ou en direct, applications coopératives diverses. Traditionnellement installés sur des réseaux locaux ou RNIS, les systèmes de vidéoconférences étaient jusqu'ici construits principalement sur des protocoles propriétaires. Leur déploiement sur internet leur impose l'utilisation de protocoles plus standardisés, afin de permettre l'interopérabilité, et de les adapter aux contraintes de l'internet. La principale caractéristique des applications multimédia est l'incorporation et la manipulation de médias continus tels l'audio, la vidéo et l'animation, pour le transfert d'information sensible aux délais sur une certaine période de temps. Le défi auquel font face ces applications est de satisfaire les exigences des usagers en fonction des possibilités du réseau utilisé. Toutes les solutions proposées par les chercheurs et celles qui restent à venir tournent autours d'un seul mot clé nommé « **Qualité de Service** » (QdS). Ce terme désigne un ensemble de propriétés, perceptibles par l'utilisateur, qui caractérisent la performance du service offert à l'utilisateur.

Actuellement, l'Internet ne peut offrir de telles garanties et offre un service «au mieux». La gestion du trafic de données est prise en charge de façon simple mais elle ne garantie ni la livraison ni la ponctualité. Pour satisfaire les contraintes temporelles induites par la manipulation de données continues, un ensemble de fonctionnalités de gestion de la QdS est nécessaire. Beaucoup de recherche en ce sens se sont focalisées sur la réservation de ressources dans des réseaux de communication comme ATM. L'application de la réservation de ressources pour Internet est encore un sujet de discussion. De nombreuses techniques basées sur la différentiation de paquets et de classes de services ont été proposées afin de fournir une certaine QdS dans l'architecture d'Internet. L'adaptation est une autre stratégie de la gestion de QdS qui permet aux applications d'être plus tolérantes envers les fluctuations de service du réseau dues à la congestion. L'adaptation de QdS est complémentaire à la réservation de ressources, et contrairement à celle-ci, est applicable à des applications évoluant sur Internet.

Dans ce mémoire, on propose une architecture qui intègre des mécanismes de contrôle et de gestion de la QdS. L'accent est mis sur les stratégies d'adaptation qui permettent à l'application de réagir à des fluctuations de services du réseau. On y propose une architecture de la gestion de la qualité de service qui englobe la définition des interfaces et les profiles des usagers. On introduit le concept de réseaux multiples, où une machine client et une machine serveur sont reliées par plusieurs réseaux offrant des QdeS différentes. Un prototype de support à une application a été développé en tenant compte des mesures de QdS. Le prototype est conforme aux standards afin d'être compatible et interopérable avec d'autres applications suivant ces mêmes normes. Nos travaux s'appuient sur le standard H.323 qui défini les spécifications pour la communication multimédias sur les réseaux type Internet.

Abstract

Using Internet to deliver Multimedia data poses a lot of challenges. One problem with Internet is it can not provide a guarantee service, but some applications like real-time applications are very sensitive to the performance of network. One important issue in this field is ***the QoS management for distributed multimedia application***.

This thesis attempted to highlight some of the issues and concerns over the QoS management for Multimedia applications. In particular we select the ***QoS adaptation*** as our study focus. The discussions are based on the concept of distributed multimedia application, QoS related concepts, and real-time multimedia application protocols.

We identify the related issues and technologies. These include QoS definitions and models, communication protocols used in multimedia application and some multimedia applications in real world.

We develop our own QoS adaptation policy and admission control strategy. The adaptation is triggered either when the system resource is scarce or more resource is available. Therefore a graceful degradation or gradually QoS increasing are used for the two situations. Depending on different usage scenarios, we propose different strategies for QoS adaptation.

We implement a prototype to integrate all proposed policies. In particular, the prototype focuses on our own QoS adaptation policy as well as the model implementation of QoS control for multimedia applications. The prototype does not depend on any particular application, thus it is general enough to be customized to most application.

The major contribution of our implementation is it provides a general model of controlling the multimedia application QoS over the IP network, which can adapt to any applications running on it.

Table of Contents

Chapter 1

Introduction

The Web has gained tremendously in popularity over the past several years, and almost everything that can be digitalized attempts to use Internet as its media carrier to increase its popularity. In recent years, there has been growing interest in implementing distributed multimedia applications over IP based network. A popular class of applications is the Internet Telephony. More and more people now days use the prepaid telephone card to make their long distance call, but perhaps only a small portion of them realizes that they are actually using the Internet to delivery voice data, one of the formats of multimedia data. The companies offering such services take advantages of the "free" data transmission of Internet to attract more customers than those telephone companies using traditional switching technologies.

IP network has been well developed in the past decade and widely accepted. IP protocol suite has been implemented well to serve the traditional data transmission with "best effort" services, which tries to utilize the network resource as much as it can. But this service does not fit in current distributed multimedia application's requirement very well, especially in the time sensitive applications like real time multimedia application. As the author points out in [Bla00], the question was not whether IP can carry voice, but a more general question of whether traffic with particular quality demands such as voice can be carried effectively and efficiently over a datagram packet data network. Again, take the prepaid telephone card as an example, most of the user's experience and suffer from such "best effort" paradigm. This is manifested as the voice delay and bad voice quality during the call.

There are two major reasons that the traditional IP network protocol cannot satisfy today's distributed multimedia applications. One is that it treats all data the same, meaning that it deploys the same policy (best effort) during the transmission of all kinds of data, no matter it is voice, text, or video. For the time sensitive data such as voice, it requires high priority to be transmitted. Related improvement technologies include resource reservation and differential services. Another problem is that multimedia applications are time sensitive but not error sensitive. Traditional IP network protocol suite, however, was designed with focus on transmitting non-error data, such as TCP protocol, but not the real-time data. Again, this does not fit into the context of the multimedia applications especially when some multimedia data are delay sensitive.

Many modifications and improvements to the IP network have been proposed since the day IP protocol suite was born. In order to justify the result of real-time related IP protocol improvement and gain experience during its deployment, many distributed multimedia application models have been developed in the research area. One important issue in this field is ***the QoS management for distributed multimedia application***.

Quality of service (QoS) plays an important role in today's multimedia data transmitting over network. In fact, the prepaid telephone card application provides a perfect example of QoS issue: you pay less to make the long distance call, but the voice quality you've got is worse than those provided by conventional telephone company.

QoS management entails effective resources management according to user's expectation. The resources include hardware resources, such as CPU cycles, memory, disk bandwidth, and network bandwidth, as well as software resources, such as access to the Web, databases, and other resources. In reality, we only have a limited (and often scarce) supply of resources. Consequently, managing the allocation and scheduling of these resources to applications is necessary to ensure that QoS expectations are met. In the case of multimedia applications, QoS management starts from the user's expectations associated with applications, based on the monitoring information, smoothly deliveries multimedia data. For our research, we use media presentation QoS requirements to make the media data transmission strategy for properly utilizing available network resource.

This thesis concentrates on QoS management issues for the multimedia applications. Corresponding to the QoS modules identified in [Cam94], we are working on a subset of its classification, more specifically, they are QoS adaptation, QoS monitoring and QoS specification. As a research result, we have implemented a general QoS management model with highlight on some functional models, it could be used to most distributed multimedia applications for QoS control. But we did not implement any specific multimedia application running on this model. This model only provides a QoS management platform. When adding some high level functions associated to some specific application, this model can be easily customized to other specific application. One of the major focuses of this model is on the QoS adaptation. We will present our own QoS adaptation strategies in the implementation part.

The purposes of this thesis are:

(1) to identify and discuss the related technologies and standards;

(2) based on the discussion, present our own QoS adaptation policy as well as the model implementation of QoS control for multimedia applications.

This thesis is divided into six main parts. The content of this thesis is organized as follows:

Chapter 2 gives a general introduction of distributed multimedia applications and some multimedia Codec libraries.

Chapter 3 covers the QoS definition and QoS management related concepts. In the summary of this chapter, one big picture of QoS management architecture is presented.

In chapter 4, the focus is on some multimedia application communication protocols, which include a bunch of IETF protocols, several RTP related IETF RFCs, ITU H.323 and networks resource management technology. The role of this chapter is to act as a refresher instead for a complete tutorial.

After introducing the distributed multimedia application related technologies and standard, Chapter 5 introduces several popular distributed multimedia applications that include both the commercial applications and research projects.

Chapter 6 is the implementation part. The implementation architecture is based on the QoS model introduced in chapter 3. It uses RTCP feedback information as the QoS monitoring information and tries to tune media transmission QoS to satisfy the network resource requirements.

Chapter 7 concludes the whole thesis and presents a few aspects of future work should be considered.

The whole project implementation is developed under Microsoft Visual C++ on Windows NT 4.0. RTP/RTCP protocol library is derived from the H.323 protocol library of Elemedia Corp [Eleme].

Chapter 2

Distributed Multimedia Application

A multimedia system is characterized by computer-controlled, integrated production, manipulation, presentation, storage and communication of independent information, which is encoded at least through a continuous (time-dependent) and a discrete (time-independent) media [Ste95].

Multimedia data is collected by analog equipment and converted to digital format through codec standards such as G.711, G.723, H.263, H.261, MPEG, AVI. Multimedia data has its own characteristics that are different from the traditional data like text and image. One difference is that multimedia data is usually time sensitive. It means we can not treat it as non-time sensitive application like E-mail: you don't care whether the e-mail can be delivered in one minute or 5 minutes. But for Multimedia application, it is quite different to play the video 5 frames per minute verses 25 frames per minute. Another difference is the word "Multi". It means more than one kinds of media type involved in the application presentation. Although these media could come from different streams but they belong to the same topic, and therefore they should be played at the same time at client site. This brings up another important issue, ***synchronization***. Before we discuss what are the new requirements introduced by the multimedia application in details, let us observe the classification of the current multimedia applications first.

2.1 Classification of the multimedia applications

In [Haf96], the author points out that the multimedia applications can be classified into presentational application, conversational application, or having both aspects. They will be summarized in the following part.

2.1.1 Presentational applications

Presentational applications take the form of Multimedia information digitally stored in one or more high capacity storage devices, such as optical storage device. User can retrieve the multimedia from the servers over a broadband network onto their display devices. Such applications include video-on-demand, news-on-demand, On-line shopping and digital libraries. The client can select the item from the title list.

Since the Presentation applications are one way direction, the critical components involved in are the server operating system and network system. One difficulty to build this kind of server includes disk bandwidth limitations on the number of streams and users to query the server at the same time. Another difficulty is the traditional operating system does not provide the real-time and continuous data delivery services. Such operations include play, pause, fast-forward, fast backward, or index search on the multimedia data, like the VCR controller. When multiple users access the server through the WAN at the same time, huge connection bandwidth to the server is also needed.

The development of the presentation applications needs both network supporting and the operating system supporting. Which include communication protocols and high speed network, real-time supporting operating system and real-time supporting storage system. In order to access and store the multimedia data efficiently, new storage system like the object database system is required.

2.1.2 Conversational applications

The Conversational application is another important multimedia application in today's Internet multimedia applications. Such kinds of applications include Video conference, IP telephony, On-line tutorial, tele-computing system, collaborate medical system. The major difference from Presentation application is the participants of Conversational applications include both client and server sites. Hence it brings more complexity to implement.

The functionalities of Conversational multimedia applications require many new supports that the traditional system can not provide. The main challenges include real-time protocol, signaling protocol, real-time operating system, high network bandwidth, reliable network service quality guarantee, real-time media capture system and so on.

Even there are lots of issues to be solved for developing the Conversational application and current IP based network is not very suitable for the multimedia application, we are pleased to see that many applications have been successfully developed. Each of those applications has met part of the requirements mentioned above. We will give more details regarding those applications in chapter 5.

2.2 Codec library introduction

When we are talking about the multimedia application, one important issue we can not forget is the *Codec*. The Codec is the storage and compression algorithm to collect the raw audio or video data and convert it into the digital files or packets, which can be transmitted by network provider and represented by end-systems. Depending on the media type to present, the codec library can be distinguished into three categories: video, audio and both of them.

2.2.1 Audio Codec

Before introducing the codec example, let us give the steps on how to convert the analog voice to coded audio data. As presented in Figure 2.1, it follows sampling, quantization and compression. The sampling is for convert the analog data to digital data with some sampling rate. Quantization is used for convert the discrete sampling data to the measurable value, e.g. 16 bits Quantization yields 65536 possible values. The last step is the compression using some kind of compression algorithm like DPCM, ADPCM, u-law [Jef99] etc.

There are tens kinds of codec for audio that have been published. We will select some popular audio codecs to introduce as follows.

(1) WAV: It is audio format defined by Microsoft for the multimedia extensions to windows. It is now the common format for the Windows platform. WAV files can store mono or stereo sound at sampling rates of up to 44K Hz (CD audio quality). Currently most WAV files use bit linear storage.

(2) AU: It is used on SUN and NeXT machine supports either 8-bit u-law encoding or 16 bit linear encoding. This format is not common on other platforms, although utilities are available that will transfer it to other formats.

(3) VOC: It is a proprietary data format defined by Creative Labs for using with their SoundBlaster family of boards.

(4) AIFF: It is a format developed by Apple and it is based on the Amiga IFF tagged file structure. AIFF is one of the formats used on Macintosh computers, and it is also used by SGI workstations.

(5) MPEG: MPEG is a working group in a subcommittee of ISO/IEC (the International Standards Organization/International Electronic Commission) that generates generic standards for digital video and audio compression. In particular, MPEG defines the syntax of low bit rate video and audio bit streams, and the operation of conformant decoders. Here is the audio standards defined by MPEG:

- **MPEG-1**: Signal channel and two channel coding at 32, 44.1 and 48 kHz sampling rate. The predefined bit rates range from 32 to 448 kbps for layer I, from 32 to 384 kbps for layer II, and from 32 to 320 bit/s for layer III.
- **MPEG-2**: The audio has Mullet-channel extension to MPEG-1 and is up to 5 main channel plus a "low frequent enhancement" channel can be coded. The bit rate range is extended up to about 1 Mbps.
- **MPEG-4**: Its audio is a very high-quality audio coding standard for 1 to 48 channels at sampling rates of 8 to 96 kHz, with multi-channel, multilingual, and multi-program capabilities. MPEG-4 provides the standard technological elements enabling the integration of the production, distribution and content access paradigms including digital television and interactive multimedia application.

(6) MP3: ISO-MPEG Audio Layer-3. It is the standard for high quality music. Typically 1 MB is equal to one minutes of music or several minutes for spoken word.

(7) G.711, G.722, G.723, G.728 and **G.729** are audio codec standard defined by ITU.

- G.711: Pulse code modulation (PCM) of voice frequencies
- G.722: 7kHz audio coding within 64 kbps.
- G.723: Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbps; designed to run at low bit rates.
- G.728: Coding of speech at 16 kbps using low-delay code excited linear prediction.
- G.729: Coding of speech at 8 kbps using conjugate structure algebraic-code-excited linear-prediction. It is well supported by many VoIP vendors.

(8) ra, ram, rpm and **rm** are multimedia file formats defined by Real-Audio and Real-Video files, which support streaming and live web broadcasting. RealMedia File Format is a standard tagged file format that users four-character codes to identify file elements. These codes are 32-bits, represented by a sequence of one to four ASCII alphanumeric characters, padded on the right with space characters. The basic building block of a RealMedia File is a chunk, which is a logical unit of data, such as a stream header or packet of data.

2.2.2 Video Codec

The video data generation includes two steps. The first step is image capture like the voice sampling in the audio to get the uncompressed raw data. Another step is compression, which is defined by specific codec standard. Figure 2.1 gives the major steps of the video data compression.

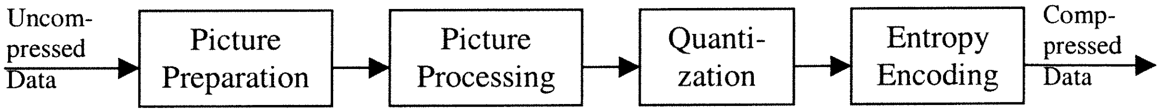


Figure 2.1: *Major steps of video compression*

1. Preparation includes analog-to-digital conversion and generating an appropriate digital representation of the information. An image is divided into blocks of 8x8 pixels, and represented by a fixed number of bits per pixel.
2. Processing is actually the first step of the compression process that makes use of sophisticated algorithms. A transformation from the time to the frequency domain can be performed using DCT (Discrete Cosine transformation). In the case of motion video compression, inter-frame coding uses a motion vector for each 8x8 block.
3. Quatization processes the results of the previous step. It specifies the granularity of the mapping of real numbers into integers. This process results in a reduction of precision. This can also be considered as the equivalence of the u-law and A-law, which apply to audio data. In the transformed domain, the coefficients are distinguished according to their significance. For

example, they could be quantized using a different number of bits per coefficient.

4. Entropy encoding is usually the last step. It compresses a sequential digital data stream without loss. For example, a sequence of zeroes in a data stream can be compressed by specifying the number of occurrences followed by the zero itself.

As for the video codec, some most popular used video Codec standards are briefly introduced as follows:

(1) Jpeg

The JPEG (*Joint Photographic Experts Group*) is a compressing standard developed by ITU, ISO and IEC. It applies to color and gray-scaled still images. A fast coding and decoding of still image is also used for video sequences known as *Motion JPEG.*

JPEG has four modes and many options. The normal way for the algorithm is as follows: (1) Break the image into 8x8 blocks and each pixel presented with 8 bits; (2) Apply DCT to each of the block; (3) Apply the quantization algorithm to wipe off less important values; (4) Reduce the (0,0) element value of each block by replacing it with the amount it differs from the corresponding element in the previous block; (5) Linearizes the 64 elements and applies run-length encoding to the list.

Since the complicated algorithm can produce 20:1 compression rate or better with good quality, JPEG is considered the ideal codec for still image.

(2) H.261 and H.263

The H.261 is an ITU standard for video conference over ISDN network. Corresponding to the term used in narrow-band ISDN standard, H.261 is also called px64. This means the H.261 stream capacity is n*64kbps (n=1,2...30), that is to say the throughput can vary from 64kbps to 1920kbps.

Unlike JPEG, H.261 defines a very precise image format. The image refresh frequency at the input must be 29.97 frames per second. During encoding, it is possible to generate compressed image rate with 10 to 15 frames per second. H.261 defines two kinds of image size. One is called *Common Interface Format* (CIF) defined by 288*352 pixels and the other is *Quarter*-CIF (QCIF) defined by 176*144 pixels. Each frame is divided into several micro-blocks, each of them has 16*16 pixels, and is divided into 6 blocks with 8*8 pixels small block.

The H.261 compression algorithm uses two different methods: intra-frame and inter-frame. In the intra-frame, there is no advantage is taken from the redundancy between frame. It is DCT based compression algorithm like JPEG. For inter-frame coding, information comparison from the previous is considered. This could correspond to the P-frame encoding of MPEG (see next paragraph).

The following characteristics is brought by the H.261standard:

- The data stream of an image includes information for error correction.
- Each image has a 5-bit temporal reference.
- The video can be display as still image by a certain command passed to decoder.

H.263 is also a video standard defined by ITU. It is based on H.261 and can provide more kinds of the data format including CIF, QCIF, sub-QCIF (128*96), 4CIF(704*576), and 16CIF(1408*1152). It also brings more efficient coding algorithm.

(3) MPEG

As we mentioned before, MPEG is defined by ISO for both video and audio. Figure 2,2 gives a brief idea how to multiplex the video and audio in MPEG [Tan96].

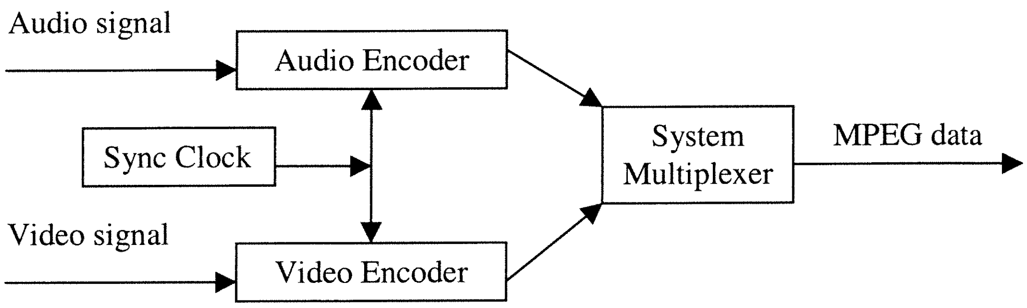


Figure 2.2: *MPEG multiplexing*

For different purposes, MPEG standard family includes MPEG-1, MPEG-2 and MPEG-4 with varied data rate. MPEG-1 produces video recorder-quality output using 1.2 Mbps. MPEG-2 which designed for SIF/CIF to HDTV at the bit rate up to 100 Mbps. MPEG-4 is for medium-resolution video conference with low frame rate (10 frames per second) and at low bandwidth (64 kbps).

MPEG can provide the VCR mode operations like fast-forward, fast-reward and random access. MPEG compression algorithm is similar to H.261 but with more aggressive motion compensation. MPEG provides four types of frame coding for processing [Tan96], which is the major difference from JPEG and make it more proper to deal with the motion picture. These frame types include:

- I (Intra-coded) frames: Self-contained JPEG-encoded still picture.
- P (Predictive) frames: Predicted picture from the previous I or P picture.
- B (Bi-direction) frames: Differences with the last and next frame.
- D (DC-coded) frames: Block averages uses for fast forward.

From discussion above, we can conclude the JPEG, H.26x and MPEG are not alternative techniques for data compression. Their goals are different and complementary. Most of the technologies are similar but not same. JPEG is proper for still image; H.26x and MPEG-4 are used for video conference at low bit rate; MPEG-2 is proper for retrieval of multimedia information like video on demand application.

2.3 Requirements of Multimedia application

After giving the multimedia application type and the codec algorithm, I would like to discuss the new requirements imposed by distributed multimedia application. We know that the participants involved in the distributed multimedia application include client, server and network. The requirements for the multimedia application come from two parts, the operating system and the network system. Issues that should be considered in operating system include CPU scheduling, memory reservation, real-time storage device management, real-time file system and DBMS supporting. For the network system, the network resource reservation, the quality of the network service guarantee, and multicast problems need to be considered. [Haf96] proposes the new requirements from communication systems and end-systems can fall into four groups: high data rate, temporal constraints, service guaranteed and communication groups. I will discuss the requirements of multimedia in the following subsection.

2.3.1 High data rate handling

Different from the traditional static data like text and image, the multimedia data is dynamic, continuous and has higher data rate. Before compression, the multimedia data rate can be up to several Gig bits per second. Such huge data throughput requires the operating system and networks have enough capacity to handle it. This requirement need higher network bandwidth, powerful end-system with real-time facilities.

2.3.2 Data collection and compression

In the interactive multimedia application, the media data need to be collected real-time. Such kind of activities, like audio sampling and video capture, are always consuming much system resource. After the data collection, the raw data will be transferred into some kind of codec by the codec library algorithm.

We have briefly introduced several codec libraries. They all used for the data compression algorithm and packing the data into frame. Different codec libraries have different algorithms, compression rates, presenting quality and used for different kinds of applications. Usually, higher compress rate with the same presentation quality will spend more memory and CPU resources. The tradeoff exists depending on which resource is scarcer. When we select the codec that uses more efficient algorithm, it will generate less stream data to transmit but consume more operating system resource. That means we use the end-system resource to trade for the network resource. Alternately, the data collection and compression can be done by the hardware if it is available. In this case, the operating system resource can be saved. You can not easily say which one is better. It does rely on the application requirements and resource availability.

2.3.3 Synchronization and real-time

The word synchronization refers to timing. Synchronization in multimedia systems refers to temporal relations between media objects in the multimedia system [Ste95]. The Synchronization requirements are divided into three categories in [Haf96], which include stream synchronization, event synchronization, and group synchronization.

Stream synchronization refers to the stream of media data transmission and presentation. From the concerning stream objects, the stream synchronization can be divided into *intra-object* synchronization and *inter-object* synchronization.

The intra-object synchronization refers to how to make different streams belong to the same topic to be presented at same time at client site. These different streams may be generated at the same time or not. One example is the presentation with slides. The speech about the slide should be played with the related slide at the same time. Video conference is another example of intra-object stream synchronization. When the synchronization is based on single stream, it is called inter-object synchronization. For example, one video frame divided into several consequent packets. How to collect all packets belong to same frame and present it is an inter-object synchronization issue.

The latency and latency jitter play very important roles in the stream synchronization. Different streams related to same topic or same stream of same resource may go along different network route to reach destination. So the latency may be different. In order to wait for the packets arriving late and hold the packets arriving early or on time, memory buffer is required in the client site. As for the real-time presentation issue, any data arriving too late that beyond some boundary will be considered as the loss data. The solution of how to deal with the unsynchronized data is another issue to be considered.

Event-based synchronization addresses what kind of actions will be taken to respond some notifications in the multimedia system. One example is the presentation windows refreshing.

Group synchronization refers to the concept "what you see is what I see". Different users belong to the same group should see the same screen and hear same voice. Such as the online-teaching application, all students joining same class should hear the same sentence at the same time.

2.3.4 Service guarantee and resource management

In order to handle the huge bit rate of multimedia data, the synchronization and real-time problem, service guarantee is required. It includes the participant of both the networks and end-systems. To provide the service guarantee, resource reservation is necessary. In order to clarify how much resource needed for service guarantee providing, the service guarantee degrees should be specified. In [Nag93], it defines the degrees of service guarantee may be distinguished by deterministic guarantee, statistical guarantee, and best-effort guarantee.

- Deterministic guarantee: It is a service guarantee that holds for every service data unit transmitted between server and client. Which means the service provided is equal or better the specified requirement.
- Statistical guarantee: This service does not promise it will satisfy each media data unit requirement. But it will provide part of the deterministic guarantee, say 80%, for some specific stream or for some period during the service established time.
- Best-effort guarantee: This service does not provide any guarantee. It tries to use the resource as efficiently as possible. It treats every data equally and tries to deal with them with the "best" effort. Current Internet protocols provide the "best effort" service.

2.3.5 Data loss control mechanisms

When the network can only provide the best effort service, the data losing can not be avoided. It is very important to make the receiver to deal with this problem and let the loss data affect the decoder presentation quality as less as possible.

2.3.6 Multicast problem

Concerning on utilizing network resource efficiently and to reduce group synchronization problems, multicast is a good solution for group communication. With the multicast, the benefit includes:

- Saving network and server resources.
- Easier to cooperate with other protocols to support real-time and synchronization.
- Saving money

In spite of the advantage of the multicast, we realize there are difficulties resist in today's network to utilize the multicast. First, each sending and receiving host's operating systems and network stack must support multicast. Second, each host's network adapter driver must implement multicast. Third, routers and switches must be multicast-capable. Fourth, the applications must be multicast-enabled.

2.4 Summary

In this chapter, I discussed the classification of multimedia applications. Because multimedia data usually involve large data size, the Codec technique is very crucial for efficiently data delivery. I also covered a brief introduction for Audio and Video Codec techniques. Last, the special requirements for multimedia applications were listed. In the next chapter, a very important concept in multimedia application, QoS, will be introduced. QoS related concepts will be given and a QoS management model will be proposed as well.

Chapter 3

QoS Definition and Related Concepts

3.1 QoS definition and deferential service

Quality of Service (**QoS**) concepts were first introduced in the performance issue of communication networks. It is not easy to give an accurate definition of QoS. Different people have different understanding, thus we could find different definitions to describe QoS in many papers and books. In my thesis, I use the definition coming from [Vog95]: "Quality of service represents the set of those quantitative and qualitative characteristics of distributed multimedia system necessary to achieve the required functionality of an application". QoS is the ability of a network element to have some level of assurance that its traffic and service requirement can be satisfied. To enable QoS requires the cooperation of all network layers from top to bottom, as well as every network element from end to end.

QoS itself does not create any network resource to increase the transmission rate, to reduce delay, and to increase the transmission correction rate. It only can provide some kind of mechanism to manage the available network resource in order to meet the application's requirement.

3.2 Why Quality of Service is required?

We know the Internet protocol is based on the simple concept that the datagrams with IP address relayed by routers to reach the destination independently. In order to utilize the network bandwidth as much as possible, it provides a kind of service called "best effort". For the traditional non-real time Internet application such as e-mail, web, and file transfer, the existing IP protocol suite works well and efficiently. With the increasing of the multimedia Internet applications, many time sensitive applications like video conference, online teaching, and Internet telephone require high bandwidth and low latency. But the traditional IP protocols treat the time sensitive stream (Audio and Video) as same as other data. So the Quality of Service is needed to serve different kinds of application requirements. In order to provide the QoS, many new protocols are developed and added to the Internet protocol suite.

We should notice that the Quality of Service is not only a terminology used in today's Internet application. It is also an important issue for the "virtual circuit" network like ATM. When a virtual circuit is established, both the transport layer and the ATM network layer must agree on a contract defining the service the ATM network will provide.

3.3 Key QoS parameters for our work

In order to specify and quantify what kind of services required by the application, the QoS parameters are used to describe each QoS requirement characteristic. In different systems and prototypes, different categories and their related parameters are identified. The parameters should be understood both by application and the network QoS manager.

The major QoS parameters related to our work are:

- **Throughout**

  It is the measure of data transmission capacity. Actually, it is a measurement of network transmission, not the characteristic required by application, even if it will affect what kind of service can be provided eventually.

- **Delay**

  It is the period of time to measure how long the data can be displayed at the receiver after generated at the sender site. It is an very important parameter when talking about the network QoS. It is a key parameter for many time-sensitive applications especially for the interactive application. It is well understood that the network throughput is a major factor that dominates the delay, but many other factors can also make big difference on it. I will discuss more details about it in Chapter 5.

- **Delay jitter**

  Because the unpredictable characteristics of the IP network, the delay of each packet may be different. The difference of each packet delay called "delay jitter". In order to handle the delay jitter, the packet receiver must reserve enough buffers to smooth the jitter.

- **Loss rate**

  It is ratio between the number of bits lost over the number of bits received as input. Since the data transmitted in packet or cell, the loss rate always presented by packet loss rate or cell loss rate.

3.4 Issues related to QoS management implementation

3.4.1 QoS Profile

The QoS Profile is used to present and record user's preference of QoS so that the service provided by the system can be tailored to different needs. Different users with different applications have different requirements, each requirement associates with some kind of network resource requirement. The advantage of using the user QoS profile is obvious. First, some kinds of QoS profile formats can help the user have a clear view of what kind of requirement should be provided to system and easy to input. Second, the formatted QoS profile makes the system easier to get information. Third, when a user opens a new application, maybe she/he wants the QoS requirement is the same as some preference she/he had input before, the QoS profile can make it easily to be reused.

When we use QoS profile to manage the QoS preference input from user, one important concern is to make it easy to understand and use. A good QoS profile management should provide less input step and can carry more QoS information. In order to achieve these goals, different types of Profile prototype have been defined. One solution is to use the multiple level QoS profile management [Alf96]. We use the multiple level QoS profile in our implementation too. For more details about our QoS profile management implementation, chapter 6 elaborates that point.

3.4.2 QoS Monitoring

The traditional IP based network provides the "Best effort" network service, which means no guarantee service. Under such paradigm, the network tries to carry the packets as many as possible. There is no mechanism that can tell the situation of the network resource and what kind of service the network can provide. Especially when we use the connectionless protocols as UDP/IP to transfer data, the network can not provide guaranteed delivery service. So the job to collect the information about the network status and make some adjustment to adapt the network resource is leaving for higher network layer.

The major role of QoS Monitoring is to collect QoS information user cared and to report it to system. The report information includes two types of information, one is coming from network and another is coming from local operating system. When system gets the report, it can decide whether the network has enough resource to transfer the data according to the user's QoS requirement or the local operating system is powerful enough to handle the data. If not, the system will try to make some adaptation to adjust data transmission rate in order to satisfy both the user's QoS requirement and network or system resource limitation.

The QoS monitoring plays a very important role in the whole QoS management system. All activities for adaptation are based on the QoS monitoring information and user's QoS requirement. In order to get accurate information about the network and system, one solution is to get the report data dynamically as frequent as possible. Unfortunately, we can not allow its happening. Because when the resource monitoring running, either for network or operating system, it itself will occupy the resource too. If the monitoring report generating frequency is too high, most of the network and system resource can not be used for the application that should be. It is a tradeoff. We have to find a balance point that the QoS monitoring can both provide accurate report and not consume too much system resource. It is really difficult to point out it in theory. Some papers have given the suggested boundary based on practical experience [Bol94].

3.4.3 QoS Mapping

As stated in the previous subsection, QoS Profile is crucial to bring connection between the user and system. But how to make the system understand the user's desire is also another important issue. The QoS mapping is used to translate the user's QoS requirements from application level to system level so that the system can understand it and to use it for comparison. For example, the user may figure out the video frame rate, frame resolution and color amount she/he wants. But the network provider can not know how to handle all these parameters. The QoS Mapping can translate them into the concept of "throughput" in order that all network service management components can understand.

Another task for QoS Mapping is used between different system levels. We know the network can be described by layer model, and each layer has its own QoS definitions and presentation ways. To translate QoS parameters from the user's application layer to some kind comparable network layer, the system should understand the difference between each pair of adjacent network layers. Theoretically, each network layer can only understand the adjacent layer QoS parameters, so the QoS Mapping should consider all differences between each adjacent network layers and translate it layer by layer. Here are some formulas to present how to calculate the major QoS parameters mapping through deferent network layers in [Boc97].

Available _ Throughput = minimum (for all i=1,…,n) of all level's throughput(i)
Delay = sum (for all i=1,…,n) of Delay(i)
Jitter = sum (for all i=1,…,n) of Jitter(i)
Log(1-Lossrate) = sum (for all i=1,…,n) of log(1-Lossrate(i))

[Haf96] indicates that there are three types of mapping. They are QoS-QoS mapping, which indicate different layer's QoS parameter translation; QoS-resource mapping, which supports mapping the QoS parameters into certain amount of source such as buffers, CPU and bandwidth; Service-system mapping, which supports mapping the services into system components.

3.4.4 QoS Adaptation

In chapter 1, we have observed the behavior of IP based network. Because the "Best effort" service does not provide guaranteed QoS, the available network resource may change in any minute. When the available network resource changes, the application data being transferred based on previous network situation will not be suitable for the desired QoS requirement. It means the system cannot satisfy previous QoS requirements, and the QoS parameters should be changed. The QoS Adaptation protocol is just proposed for this purpose to tune to the new QoS parameters that can be accepted by user and systems.

The QoS Adaptation is triggered by the event called ***QoS violation***, it usually happens after current network and operating system resource can not provide the QoS as they can make before. It follows the QoS policing to make a QoS degradation path, which comes from the information provided by user's QoS profile and QoS monitoring. Each node in this path is a kind of agreement on some degree of QoS and resource availability. The system tries to find a balance point that is nearest to user's desired QoS under the available network and operating system resource.

We should notice that QoS adaptation has two directions: degrade and upgrade. This means it not only provides a degradation path when QoS violation happens but also provides some kind of mechanism to try to increase the QoS. The second case happens when it finds the current required QoS can always be satisfied and the extra network or system resource is still available. Under this situation, the QoS adapation should try to use the idle network or system resource to improve/upgrade the QoS if current QoS is not as good as user's desired. When increasing the QoS, the system also follows some kinds of QoS policing, which is the same as the QoS decreasing. Such policing is determined by the information coming from the user's QoS Profile and QoS monitoring information as well.

As just mentioned, there are two different directions for adaptation. We also pointed out these two adaptation directions rely on the information coming from the user QoS profile and QoS Monitoring. The system tries to decrease the QoS when QoS violation happens but to increase the QoS when QoS violation does not happen. It means system tries to improve QoS when user application is still running on some "good" QoS situation, but decreasing QoS happens when the application can not work well because of the "Bad" QoS.

From the fact that QoS adaptation itself will spend network and operating resource, it is not worth to increasing QoS when the application working well in current QoS situation. But it is necessary to decrease QoS immediately when application can not work well (QoS violation happens). So, the **reaction speed** for increasing QoS and decreasing QoS should *not* be the same. You will find more details about these kinds of reaction speed in our implementation.

3.4.5 QoS Negotiation

The role of QoS Negotiation is to find an agreement on QoS parameters between the systems and application request. The cost is an important parameter in the QoS Negotiation procedure. If it is not included in the QoS Negotiation protocol, the user will always ask for the best available QoS. Each time during the QoS negotiation procedure, the system will give a notification to user whether the user's request is admitted or not. If user's request is admitted, the notification information will carry what kind of QoS the system will provide and how many users will be charged. After the user receiving notification, he can accept it or reject. In [Haf96], it gives the three options: (1) accept the cost and the services provision starts, (2) reject the cost and abandon the service, or (3) initiate a new negotiation to relax his/her QoS requirement.

3.4.6 Admission Control

The task of Admission Control is to compare the application QoS requirement with the available system and network resource to make the judgment whether the new requirement be admitted or not. It is also used to decide whether the adaptation activity is proper after QoS adaptation being executed as well.

When a new requirement arrives, the Admission Control will run a set of test to determine whether remaining session's QoS will be violated after adding the new session. It will make a bunch of comparisons, both of network resource and operating system resource. If both of them can satisfy application's request and does not affect QoS of other running sessions, the successful signal will be issued. After one Admission Control successful signal issued, the new session information will be added into the system lookup table. In the multicast supported network, the new group list containing the new session information will be broadcast as well.

To make the comparison, the Admission Control should get the information both from the system resource and the application QoS request. We know the system resource include two kinds of information, one from operating system another come from network. For a new session, the Admission Control module has no idea how much system resource available for it, especially the network resource along the new session transmission path. Therefore, it is necessary to run a small test for collecting the network resource information. If we use the Admission Control for the QoS adaptation admission purpose, the test is not necessary. Because some real application data has been transmitted, and we have get the information from QoS Monitoring report. We can assume the network situation is the same as before and use the previous network resource information to make the comparison for future admission control.

3.4.7 Resource Reservation

When we mention QoS, we should talk about Resource Reservation. We have pointed the resource includes two types, one is local operating system resource and another is the network resource. I will talk about the network resource first.

In the traditional IP based network, the "best effect" network does not provide guaranteed service. That means there is no resource reserved for any special application when it requires this service. We know in ATM network, it can provide guaranteed QoS service by resource reservation. The ATM network use virtual circuit network to make the connection first, the reserved virtual circuit is dedicated to the application even sometimes there is no data being transmitted. But the traditional IP based network can not work in this way, each node from original address to the destination address has no relationship with each other. There is no connection established. After relaying one packet from one application, the router does not know when it will serve for the same application next time. It is really difficult to make the IP based connectionless network to provide guarantied QoS service by the traditional IP protocol suite. Fortunately, the new protocol like RSVP [Brb97] has been proposed. With RSVP, to reserve the IP network resource became possible. The protocol needs all routes along the packet transmission path running RSVP.

The resource reservation does not follow the idea of "best effort", which will use the network as much as possible. On the contrary, Resource Reservation will sometimes waste the network resource. But if the user has certain requirements and agrees to pay more for this kind of service, why not make it happen?

Because of the characteristic of the multimedia application, real-time, end system must have enough power to handle the data. So the reservation of local resource is needed. The end system resource includes CPU slots, buffer space, memory, bus bandwidth and so on. The mechanism to reserve such kind of resource can be implemented by operating system call function. In [Gop95], it proposed a mechanism for operating system task scheduling.

3.5 QoS adaptation model

Recent research has proposed many QoS management architecture models. In [Laz96], it gives architecture of "*xbind*" [Xbind], which is a multimedia application programming platform for creating, deploying and managing sophisticated multimedia services. In [Cam94], it presents an object-oriented QoS architecture named "QoS-A", which covers both the dataflow control and QoS interface.

When putting all the function modules introduced in this chapter, we can get a big picture of the QoS Management Model, as showed in Figure 3.1. In the figure, it presents the general relationships between the QoS modules mentioned in previous section. The major parts in this model include the QoS monitoring, QoS adaptation, QoS Mapping, User QoS Profile, Admission control and Resource management.

In this model, we could find the QoS Manager plays the hardcore role of the QoS management, which is made up of QoS adaptation, QoS negotiation and Admission control. Almost all information, both from user and the system, is transmitted to the QoS Manager. The information includes User's QoS preference information translated by QoS Mapping module, the QoS monitoring information and available resource report. The control message is also issued from QoS Manager to tell other module what to do in the next step. Such information could comprise of resource management control message and media sending adjustment message to Streamer.

Before sending control message to the Resource management module and the Streamer, the QoS adaptation, QoS negotiation and Admission control modules will work together to decide what kinds of control message to be issued. These control messages are based on the information got form user and system and followed some kind of adaptation and negotiation strategy.

In our implementation part (chapter 6), we will refer this model again to build the framework of our prototype and we will give more details about the QoS management. The adaptation strategy will be discussed in chapter 6 as well.

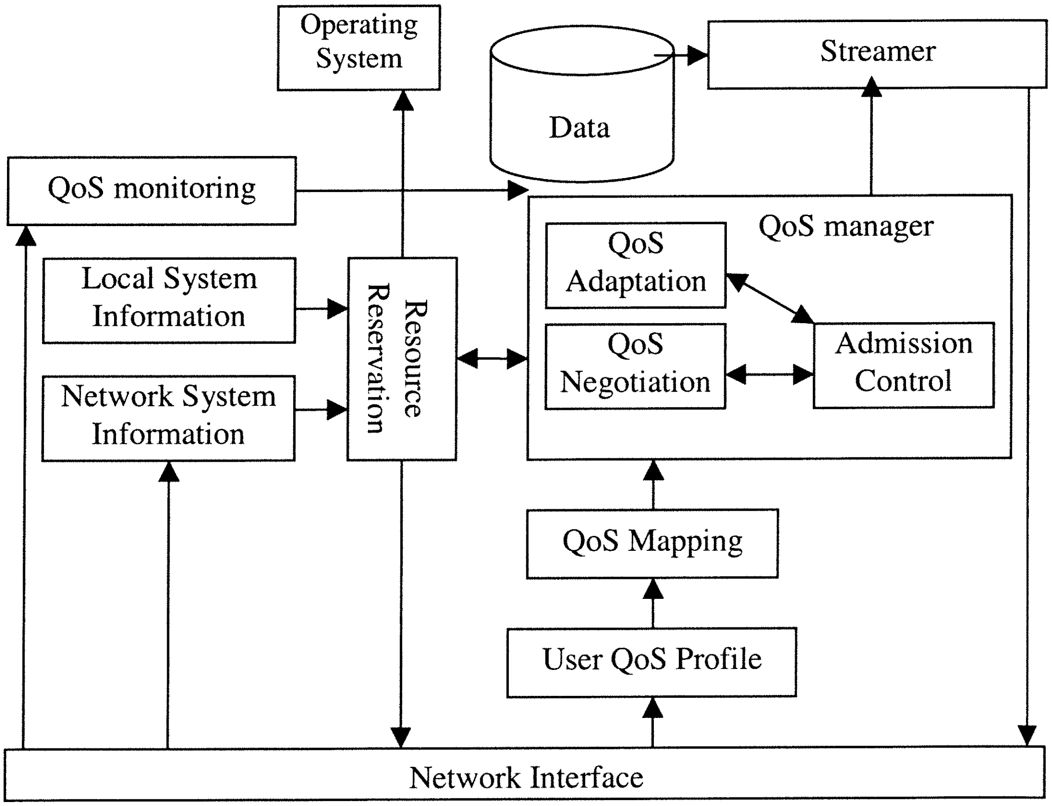


Figure 3.1: *Big picture of QoS Management model*

3.6 Summary

In this chapter, QoS, a very important concept in distributed multimedia applications, and its related issued were discussed. Those functional model used for our project were also introduced. Then, we proposed the QoS management model, which is a guideline of our project implementation. When it comes to implementation, we have to discuss the underlying network protocol and see whether and how they are designed to support the QoS. Therefore, in the next chapter, we will investigate the communication protocols related to the multimedia applications.

Chapter 4

Communication Protocols

Related to Multimedia Applications

When we talk about the distributed multimedia applications, the network is a crucial component. Most multimedia applications require high bandwidth and real-time delivery, but the current status of the service provided by the network is "best" effort and network resource is limited. Therefore, many protocols have to be added on top of today's IP-based network to satisfy requirements imposed by distributed multimedia applications. This chapter introduces these relevant protocols.

4.1 IP based real time related protocol suites

Most protocols used for multimedia applications have been proposed in the form of IETF RFC or IETF INTERNET-draft. Only a few of them are covered in this part, more complete illustration can be found in [IETF]. One exception is SIP protocol, which will be discussed in the next subsection in order to make comparison with its counterpart protocol - H.323.

4.1.1 RTP/RTCP

The real-time transport protocol (RTP) provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. Those services include payload type identification, sequence numbering, time stamping and delivery monitoring. Applications typically run RTP on top of UDP to make use of its multiplexing and checksum services. However, RTP itself does not provide any mechanism to ensure timely delivery or provide other QoS guarantees, but may rely on other suitable underlying network or transport protocols. The protocol running over RTP may utilize the information brought by RTP to make the strategy to provide QoS. We will give some multimedia application examples using RTP in the next chapter. RTP also supports data transfer to multiple destinations using multicast distribution if provided by the underlying network.

RTP protocol consist two closely-linked parts:

- The real-time transport protocol (**RTP**), to carry data that has real-time properties.
- The RTP control protocol (**RTCP**), to monitor the quality of service and to convey information about the participants in an on-going session.

4.1.1.1 RTP packet format [Sch96]

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

V=2	P	X	CC	M	PT	Sequence number
Timestamp						
Synchronization source (SSRC) identifier						
Synchronization source (SSRC) identifier						
Contributing source (CSRC) identifiers						
....						
header extension						
Payload (audio, video, ...)						

Figure 4.1: *RTP packet format*

- **version (V):** (2 bits) This field identifies the version of RTP. The version defined by this specification is 2.
- **padding (P):** (1 bit) If the padding bit is set, the packet contains one or more additional padding octets at the end which are not part of the payload. The last octet of the padding contains a count of how many padding octets should be ignored. Padding may be needed by some encryption algorithms with fixed block sizes or for carrying several RTP packets in a lower-layer protocol data unit.
- **extension (X):** (1 bit) If the extension bit is set, the fixed header is followed by exactly one header extension.
- **CSRC count (CC):** (4 bits) The CSRC count contains the number of CSRC identifiers that follow the fixed header.
- **marker (M):** (1 bit) The interpretation of the marker is defined by a profile. It is intended to allow significant events such as frame boundaries to be marked in the packet stream. A profile may define additional marker bits or specify that there is no marker bit by changing the number of bits in the payload type field.
- **payload type (PT):** (7 bits) This field identifies the format of the RTP payload and determines its interpretation by the application. A profile specifies a default static mapping of payload type codes to payload formats. Additional payload type codes may be defined dynamically through non-RTP means.
- **sequence number:** (16 bits) The sequence number increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence.
- **timestamp:** (32 bits) The timestamp reflects the sampling instant of the first octet in the RTP data packet. The sampling instant must be derived from a clock that increments monotonically and linearly in time to allow synchronization and jitter calculations.
- **SSRC:** (32 bits) The SSRC field identifies the synchronization source. This identifier is chosen randomly, with the intent that no two synchronization sources within the same RTP session will have the same SSRC identifier.
- **CSRC list:** (0 to 15 items, 32 bits each) The CSRC list identifies the contributing sources for the payload contained in this packet. The number of

identifiers is given by the CC field. If there are more than 15 contributing sources, only 15 may be identified. CSRC identifiers are inserted by mixers, using the SSRC identifiers of contributing sources.

4.1.1.2 RTCP Packet Format

The RTP control protocol (RTCP) is based on the periodic transmission of control packets to all participants in the session, using the same distribution mechanism as the data packets. RTCP does not transfer media data, just the control information related to an RTP data stream. RTCP messages are transmitted to the same IP address as the corresponding RTP data stream but with another port number. Adaptive application frequently uses the sender and receiver reports to adapt their transmissions to current network conditions. When packet loss rates increase and reach some bound, the sender can decrease their transmission rates to decrease the data loss rate. When the data loss rate decreases, the sender can switch back to more bandwidth-consuming codec.

Each RTCP packet contains a number of elements, usually a sender report (*SR*), or receiver report (*RR*) followed by source descriptions (*SDES*). We give the major message type in RTCP as follows:

- ***SR*** (Sender report) is generated by user who also sending RTP packet. It describes the amount of data sent so far, also send RTP timestamp and absolute time for synchronization.
- ***RR*** (Receiver report) is sent by user who receive RTP packet. It describes the packet loss rate and jitter from the source. It also indicates the last timestamp and delay since receiving a send report.
- ***SDES*** (Source description) is source description items. It contains CNAME, a globally unique identifier similar in format to an email. It also identifies the participant through its name, e-mail and phone number.
- ***BYE*** is used for a user's leaving.
- ***APP*** is used to add application-specific information to RTCP packets.

Since the sender reports, receiver reports and *SDES* packets contain information that can continually change, it is necessary to send these packets periodically. In order to prevent flood of RTCP reports in a large RTP session group, some mechanism have been used to ensure the bandwidth used for RTCP reports remain fixed, independent of the group size.

4.1.1.3 Payload formats

RTP allows for payload formats to be defined for each particular codec. These payload formats describe the syntax and semantics of the RTP payload. The particular semantic of the payload is communicated in the RTP payload type indicator bits. Lots of standard encoding and their payload types can be used within RTP by profile specification. More details can be found in [Sch96b]. Further more, any one can register a new codec name and procedure is defined for doing so. This allows for RTP to be used with any kind of codec developed by anyone.

4.1.2 Some additional adjustments for RTP

RTP/RTCP is widely used in distributed multimedia applications. Many researchers have worked on it and proposed lots of additional advice on the basic RTP definition, which improve the efficiency and utilization of RTP. Many of the ideas are presented in IETF RFC documents.

RTP payload for redundant audio data

Due to unreliability of UDP transport, one of the most significant problems of RTP transmission is packet loss. The addition of redundancy to the data stream provides a solution to solve the loss data recovery. If a packet is lost then the missing information may be reconstructed at the receiver from the redundant data that arrives in the following packet(s), provided that the average number of consecutively lost packets is small. For more details, please see [Per97].

RTP Payload Format for Generic Forward Error Correction [Ros99]

This format specifies a payload format of RTP that allows for generic forward error correction of real time media. It uses the FEC algorithm to recover the loss data that include both the payload and the critical RTP header fields. In its specification, it provides much flexibility of how to use the FEC algorithm.

RTP payload format for user multiplexing [Sch98d]

In order to present how the RTP multiplexing works and its advantage and disadvantage, we use an example of Voice over IP (VoIP) application. In this example, we introduce a new terminology, *Internet telephone gateways* (ITGs). It allow a public switched telephony user (PSTN) user to contact another PSTN user, with the long distance portion of the call routed over the Internet. The VoIP RTP data multiplexing is depicted in Figure 4.2.

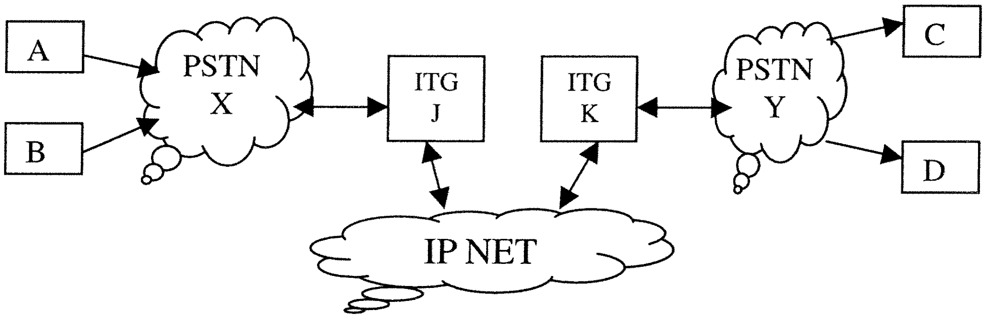


Figure 4.2: *RTP multiplexing of* ***VoIP***

Subscribers A and B connect to ITG J via their local telephone network X. A wishes to speak with user C, and B wishes to speak with user D, both of which are connected to local phone network Y. To complete the call, ITG J packetizes and transports the voice from A and B to ITG K through the IP network. After getting the data, ITG K completes the calls to C and D through PSTN Y. This type of arrangement and common destination may be particularly common for connecting the PBXs of corporate branch offices across the Internet. In this scenario, media data is transported via a separate RTP session for each user. We observe that using a separate RTP session for each user connected between a pair of gateways is wasteful.

The pros and cons using RTP multiplexing are:

- The first goodness for multiplexing is it can increase efficiency evidently. Typically, the multiplexing protocol definition only adds 16 bits of overhead per multiplexed user, while the payload data increasing is much more than that. In VoIP, since most voice trunks can carry at least 24 calls at a time, we can imagine how much efficiency it will bring.
- A further benefit of multiplexing is a potential reduction in packetization delays. Most distributed multimedia application use fairly large packetization delays, mainly for the purpose of raising the size of the payloads to increase efficiency.
- Another benefit is the reduction in interrupt processing at the receiver. Whenever a packet arrives at the gateway, the operating system must perform a context switch into the kernel and process the packet.
- The main drawback of multiplexing is the increase in store-and-forward delays. These delays are often cared about most in end systems, which are typically connected via dialup modems.

RTP Payload Format for specific codec

RTP protocol has been widely accepted as the multimedia transmission protocol. Many proposals for specific codec standard have been given and discussed. In the following, we give a few examples that published as IEFT RFC.

- ***RTP Profile for Audio and Video Conference with Minimal Control*:** [Sch96b] it provides interpretations of generic fields within the RTP specification suitable for audio and video conference. It defines a set of default mapping from payload type numbers to encoding. It also describes how audio and video data may be carried with RTP.
- ***RTP Payload Format for H.261 and H.263*:** [Tur96] [Chu97] Based on some experience on multimedia conference application, the standard for carrying H.261 and H.263 video data over FTP packet have been proposed. In these RFC, they pointed out how to packetize the H.261 (H.263) packet according to H.261

(H.263) standard definition, how to solve the packet loss problem, how to compose and send the control packets.

- Other codec standards described in RFC for RTP payload format also include MPEG [RFC2250], JPEG [RFC 2345], sun CellB video coding [RFC 1023] and so on.

4.1.3 RSVP

The *resource reservation protocol (RSVP)*[Brb97] is used by a host to request specific qualities of service from the network for particular application data streams or flows. RSVP is designed to operate with current and future unicast and multicast routing protocols.

In order to accommodate large groups efficiently, dynamic group membership, and heterogeneous receiver requirements, RSVP makes receivers responsible for requesting a specific QoS. A QoS request from a receiver host application is passed to the local RSVP process. The RSVP protocol then carries the request to all the nodes (routers and hosts) along the reverse data path(s) to the data source(s), but only as far as the router where the receiver's data path joins the multicast distribution tree.

Quality of service is implemented for a particular data flow by mechanisms collectively called "traffic control". These mechanisms compose a packet classifier, admission control, and a "packet scheduler" or some other link-layer-dependent mechanism to determine when particular packets are forwarded.

In [Brb97], RSVP has the following attributes:

- RSVP makes resource reservations for both unicast and many-to-many multicast applications, adapting dynamically to changing group membership as well as to changing routes.
- RSVP is simplex, i.e., it makes reservations for unidirectional data flows.

- RSVP is receiver-oriented, i.e., the receiver of a data flow initiates and maintains the resource reservation used for that flow.
- RSVP maintains "soft" state in routers and hosts, providing graceful support for dynamic membership changes and automatic adaptation to routing changes.
- RSVP is not a routing protocol but depends upon present and future routing protocols.
- RSVP transports and maintains traffic control and policy control parameters that are opaque to RSVP.
- RSVP provides several reservation models or "styles" (defined below) to fit a variety of applications.
- RSVP provides transparent operation through routers that do not support it.
- RSVP supports both IPv4 and IPv6.

RSVP is a good candidate for QoS management over the IP based network, but it is not widely used till now. The major reason is it needs all routers along the transmission path support RSVP protocol that is not possible in current Internet architecture.

4.1.4 RTSP

The Real-Time Streaming Protocol (RTSP) establishes and controls either a single or several time-synchronized streams of continuous media such as audio and video. It does not typically deliver the continuous streams itself, although interleaving of the continuous media stream with the control stream is possible. In other words, RTSP acts as a "VCR remote control" for distributed multimedia applications. It allows a client to instruct a media server to record and playback multimedia sessions, including functions such as seek, fast forward, rewind and pause.

There is no notion of an RTSP connection; instead, a server maintains a session labeled by an identifier. An RTSP session is not tied to a transport-level connection such as a TCP connection. During an RTSP session, an RTSP client may open and close many reliable transport connections to the server to issue RTSP requests. Alternatively, it may use a connectionless transport protocol such as UDP.

The streams controlled by RTSP may use RTP, but the operation of RTSP does not depend on the transport mechanism used to carry continuous media. RTSP is a textual protocol similar in format to HTTP so that extension mechanisms to HTTP can in most cases also be added to RTSP.

RTSP supports the following operations [Sch98c]:

- **Retrieval of media from media server**: The client can request a presentation description via HTTP or some other method. If the presentation is being multicast, the presentation description contains the multicast addresses and ports to be used for the continuous media. If the presentation is to be sent only to the client via unicast, the client provides the destination for security reasons.
- **Invitation of a media server to a conference**: A media server can be "invited" to join an existing conference, either to play back media into the presentation or to record all or a subset of the media in a presentation. This mode is useful for distributed teaching applications. Several parties in the conference may take turns "pushing the remote control buttons."
- **Addition of media to an existing presentation**: Particularly for live presentations, it is useful if the server can tell the client about additional media becoming available.

4.1.5 SDP

The purpose of SDP (Session Description Protocol) is to convey information about media streams in multimedia sessions to allow the recipients of a session description to participate in the session [Jac98], but not for describing the media encoding. This information can be used for other protocols or applications, such as SIP. The SDP session description is entirely text formatted using the ISO 10646 character.

The SDP includes [Jac98]:

- Session name and purpose
- Time(s) the session is active
- The media comprising the session
- Information to receive those media (addresses, ports, formats and so on)

Some additional information may also be desirable:

- Information about the bandwidth to be used by the conference
- Contact information for the person responsible for the session

In general, SDP must convey sufficient information to be able to join a session (with the possible exception of encryption keys) and to announce the resources to be used to non-participants that may need to know.

4.2 H.323 and SIP

We have introduced the RTP protocol and related technologies. All this protocols only provide how to transmit media data and how segment frame to compose the packet. In this section, two major protocols, H.323 and SIP, will be discussed, with the focus on the call signaling for connection establishment, capabilities exchange and conference control.

4.2.1 SIP

The *Session Initiation Protocol (SIP),* developed in the MMUSIC working group of the IETF, is an application-layer control protocol that can establish, modify and terminate multimedia sessions or calls. These sessions include multimedia conferences, distance learning, Internet telephony and similar applications consisting of one or more media types as audio, video, white board etc. SIP is a client-server protocol. Participants can be a human user, a "robots" (media server) or a gateway to another network. SIP can invite parties to both unicast and multicast sessions; the initiator does not necessarily have to be a member of the session to which it invites.

Media and participants can be added to an existing session. SIP makes minimal assumptions about the underlying transport protocol. Therefore, it does not matter whether the underlying protocol is UDP or TCP.

SIP transparently supports name mapping and redirection services. These facilities also enable "personal mobility", the ability of end users to originate and receive calls and access subscribed telecommunication services on any terminal in any location, and the ability of the network to identify end users as they move [Han99].

SIP supports five facts of establishing and terminating multimedia communications [Han99]:

- **User location:** determination of the end system to be used for communication.
- **User capabilities:** determination of the media and media parameters to be used.
- **User availability:** determination of the willingness of the called party to engage in communications.
- **Call setup:** "ringing", establishment of call parameters at both called and calling party.
- **Call handling:** including transfer and termination of calls.

SIP Components

There are two components in a SIP system, user agents and network servers. A user agent is an end system that acts on behalf of a user. Usually it consists of two parts, a client and a server. The client part, User Agent Client (UAC), is used to initiate a SIP request. The server part, User Agent Server (UAS), receives requests and returns responses on behalf of the user.

There are two kinds of network servers, proxy servers and redirect servers. Two examples corresponding to types can be found in section 3.5.3. A SIP proxy server forwards requests to the next server after deciding which it should be. Such proxy can be any kinds of SIP server. Before the request reached the UAS it may have traversed several servers. Also the response will be traversed in reverse order. As a proxy server issues both requests and responses, it acts as both a client and a server role. The other network server is redirect server. It does not forward requests to the next server. Instead it sends a redirect response back to the client containing the address of the next server that the client should contact.

SIP messages

There are two kinds of SIP messages, requests and responses. Unlike other signaling protocols such as H.323, SIP is a text-based protocol. This makes a SIP header largely self-described and minimizes the cost of entry. It also leads to simple parsing and generation, particularly when done with powerful text processing language [Sch98b].

Request messages in the current version of SIP (version 2.0) [Han99]:

- **INVITE:** Indication of the user or service being invited to participate in a session.
- **ACK:** Confirmation that the client has received a final response to an INVITE request.
- **BYE:** The user agent client uses BYE to indicate to the server that it wishes to release the call.
- **CANCEL:** The CANCEL request cancels a pending request, but does not affect a completed request. (A request is considered completed if the server has returned a final status response.)
- **OPTIONS:** It queries information about capabilities, but does not set up a connection.
- **REGISTER:** A client uses the REGISTER method to register the address listed in the To header field with a SIP server.

Response messages: After receiving and interpreting a request message, the recipient responds with a SIP response message, including the status of the server, success or failure. The responses can be of different kinds of response identified by an ID code, a 3-digit integer. The first digit defines the class of the response.

An example of SIP location server

The most important SIP operation is inviting new participant to a call. A SIP client first obtains an address where the new participant is to be contacted, of the form name@domain. The client then tries to translate this domain to an IP address where a server may be found. The server who receives the message is not likely to be the user agent server where the user is actually located; it may be a proxy or redirect server. Figure 4.3 shows the behavior of the SIP redirect server.

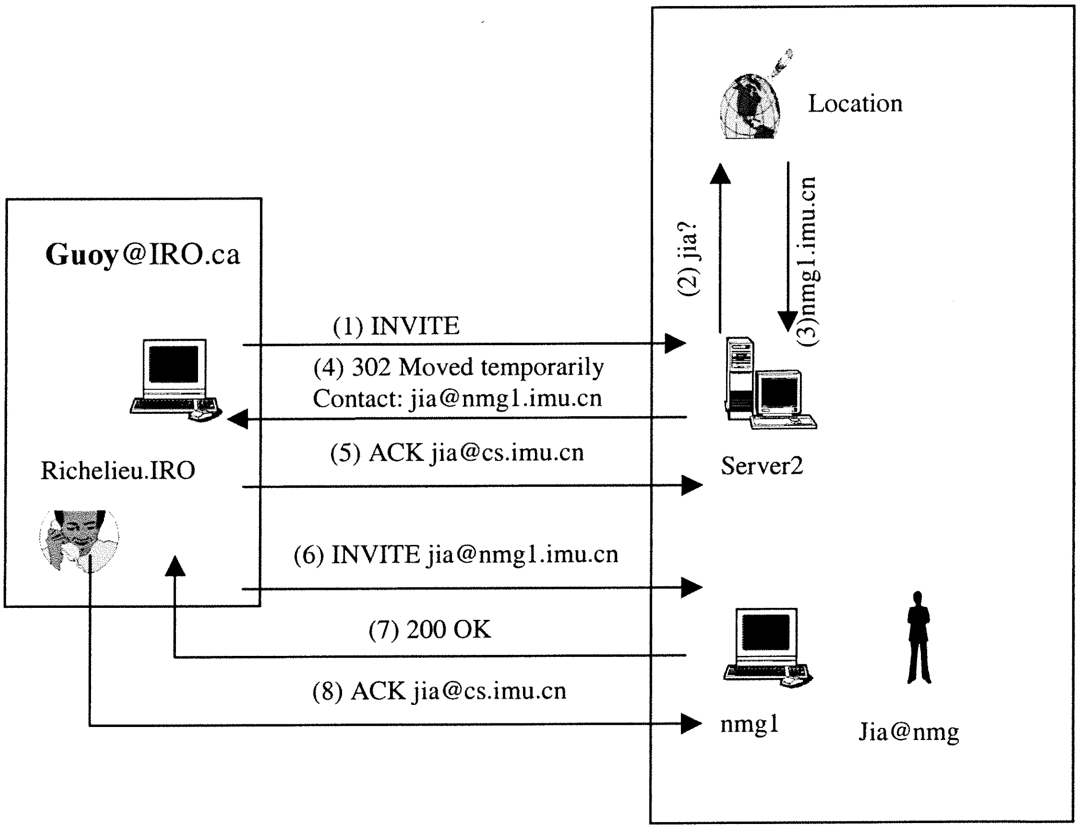


***Figure 4.3:** Example of SIP redirect server*

4.2.2 H.323

H.323 is an umbrella recommendation from the International Telecommunications Union (ITU) that sets standards for multimedia communications over Local Area Networks (LANs) that do not provide a guaranteed Quality of Service (QoS)

[Kra98]. H.323 stared out a protocol for multimedia communication on a LAN segment, but has evolved to try and fit the more complex needs of Internet telephone. The H.323 standard provides a foundation for audio, video, and data communications across IP-based networks.

The H.323 Version 2 specification was approved in January 1998. The standard is broad in scope and includes both stand-alone devices and embedded personal computer technology as well as point-to-point and multi-point conferences. H.323 also addresses call control, multimedia management, and bandwidth management as well as interfaces between LANs and other networks.

4.2.2.1 H.323 network elements

The H.323 recommendation defines a number of components. We list the four major components for a network-based communications system as follows.

- **Terminal:** H.323 terminals are one of the several initial input/output devices of the VoIP services. The terminal uses an audio codec to encodec/decode audio waves into audio frames that can then be encapsulated into IP packets and routed to another H.323 entity. Also the terminals have the H.323 signaling stacks (H.225.0, H.245) that is used for establishing and maintaining the VoIP calls.
- **Gateway:** An H.323 Gateway is an endpoint which provides translation function for making real-time, two-way communications between H.323 terminals on an IP network and other ITU terminals; phones on the PSTN; other terminals on other networks.
- **Gatekeeper:** Gatekeeper is an important H.323 entity that provides address translation and controls access to the network for H.323 terminals and gateways. The gatekeeper may also provide other services to the H.323 terminals and gateways, such as bandwidth management and locating gateways.
- **MCU (Multi-point control Units):** The MCU supports conferences between three or more endpoints. Under H.323, an MCU consists of a Multi-point Controller (MC), which is required, and zero or more Multi-point Processors (MP). The MC handles H.245 negotiation between all terminals to determine

common capabilities for audio and video processing. The MC also controls conference resources by determining which, if any, of the audio and video streams will be multicast.

4.2.2.2 H.323 protocol stack

H.323 protocol stack is composed by a group of protocols. It includes H.245 for negotiating channel usage and capabilities, H.225.0 for connection establishment, Q.931 defined in H.225.0 for setup and tear down H.323 session, T.120 for data conference, H.261 and H.263 for video, G.711, G.723 and G.729 for audio, and RAS (Registration, Admission and Status) defined in H.225.0 for communication with Gatekeeper.

H.323 Protocol stack

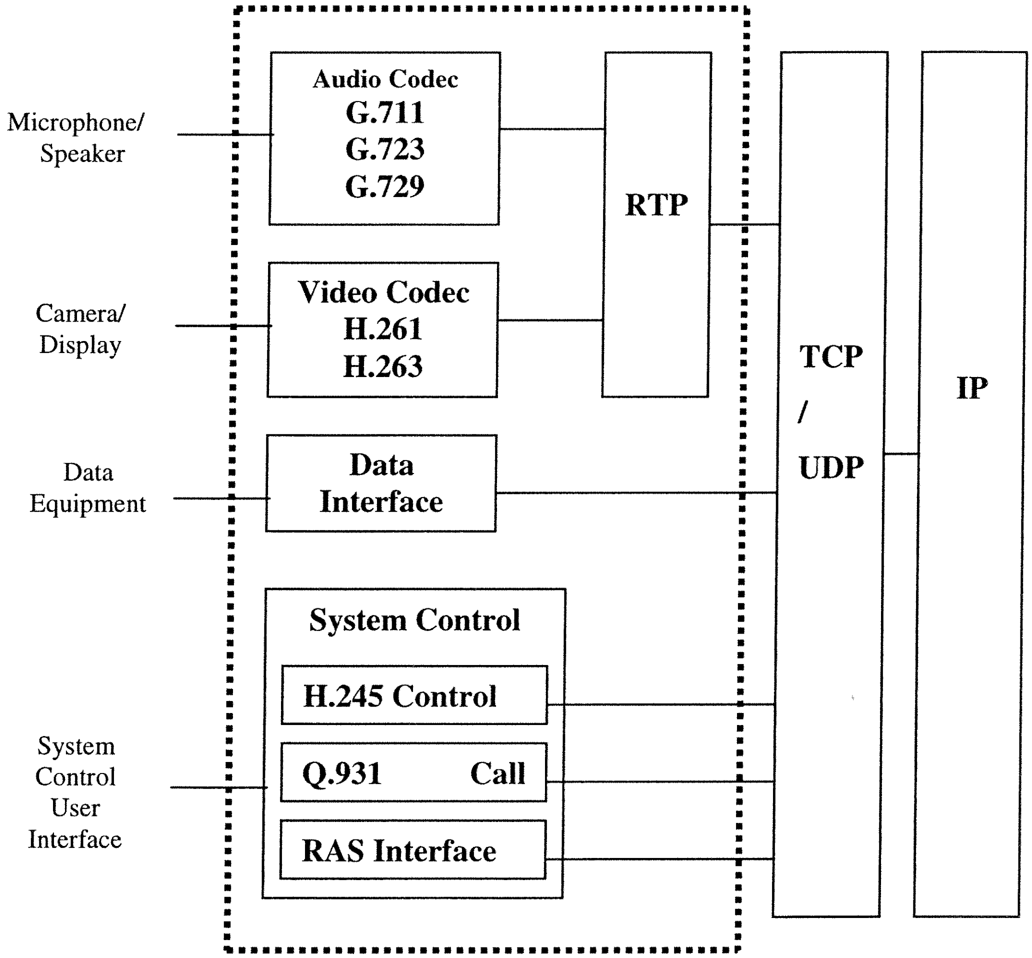


Figure 4.4: *H.323 protocol stack architecture*

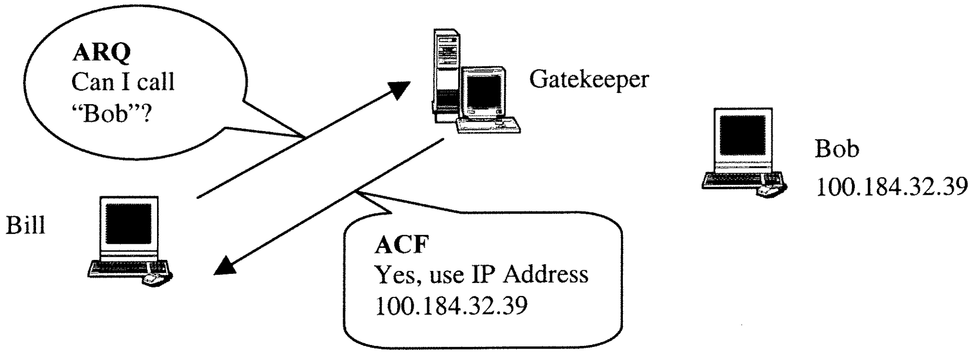
H.323 uses both reliable (TCP) and unreliable (UDP) transmission protocols for all protocols mentioned above. The unreliable protocol can provide higher speed, but no delivery guarantee. H.323 uses TCP for H.245 control channel, T.120 data channels and Q.931 call signaling channel, while using UDP for audio, video and RAS channel. The audio and video streams in H.323 use RTP on top of UDP for transport. The relationship of these protocols is described as Figure 4.4.

4.2.2.3 H.323 Call Processing

From the protocols described above, we could find the H.323 signal protocol is really complex. One reason is that H.323 borrows many recommendations from other ITU standards like H.320 for videoconference over ISDN. In order to understand H.323 call procedure, a sample of H.323 call scenario is described and illustrated below.

Basically there are divided into two steps:

- Figure 4.5 is an UDP based address resolution and permission step.
- Figure 4.6 is the TCP based messaging, which establishes the call control circuit and subsequent media channels between two or more participants.



ARQ: Admission Request
ACF: Admission Confirm

Figure 4.5: *H.323 Gatekeeper Messaging*

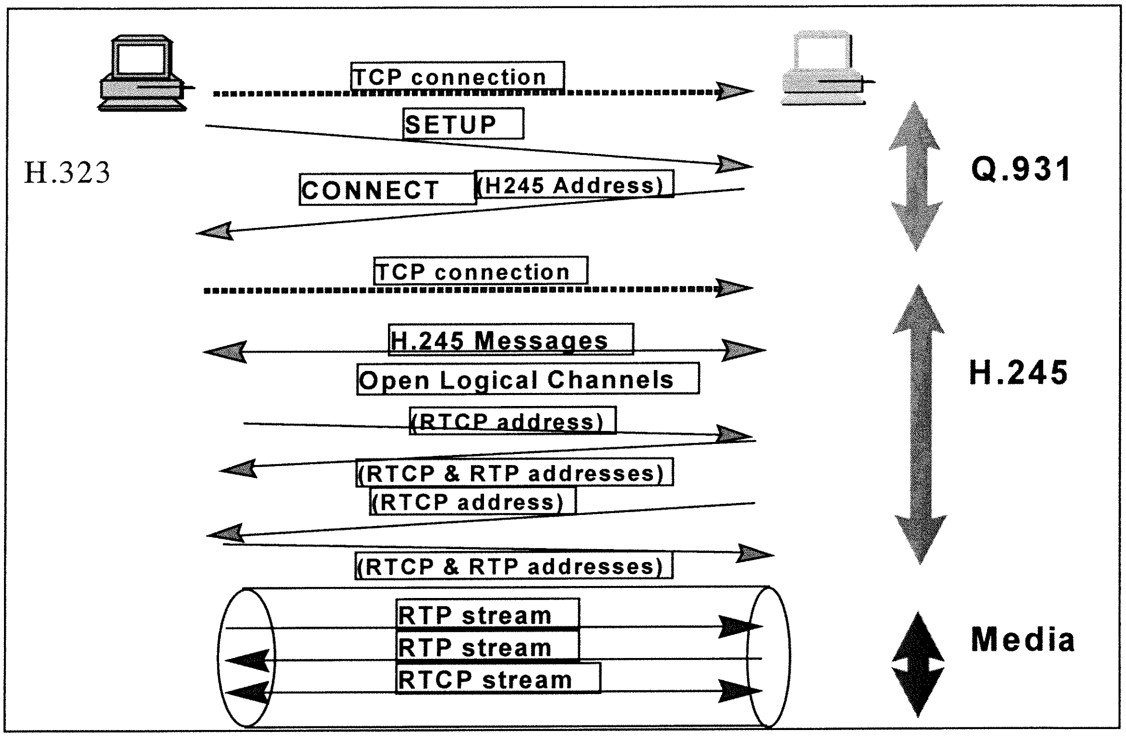


Figure 4.6: H.323 Message Flow

4.2.3 SIP *vs.* H.323

H.323 protocol is widely used in current commercial field. But we also notice that SIP begins to be supported in the industry because of its own characteristics. In this section, we will give the comparison based on the paper [Sch98b]. We know H.323 is a series of protocols not limited on the signaling control, but includes media codec and RTP implementation. Our comparison is only focus on signaling control because of the functionality of SIP.

- **Complexity:** H.323 is the most complex of these two protocols. It uses a binary representation for its messages, which is based on ASN.1 and the packed encoding rules (PER). ASN.1 generally requires special code-generator to parse. SIP, on the other hand, encodes its messages as text, similar to HTTP. This leads to simple parsing and generation, particularly when done with powerful text processing languages such as Perl. This textual encoding simplifies debugging, allowing manual entry and analyzing of messages. Because H.323 borrows many instructions from other ITU standard, it makes a single request need interact between several protocol components. While SIP only issue a simple command.

- **Extensibility:** SIP has learned the lessons from HTTP and SMTP and built in a rich set of extensibility and compatibility functions. There are six basic classes in SIP, and each of them is identified by the hundreds digit in the response code. Because of the textual encoding, the header field is self-describing. Which means it is very easy to add new header. The developers only determine usage of the name and add support for the field. H.323 provides extensibility as well. These are generally nonstandard Param fields placed in various locations in the ASN.1. These parameters contain a vender code, following by an opaque value that has meaning only for that vendor. In H.323, each codec must be centrally registered and standardized. While in SIP, it uses SDP to convey the codec supported by an endpoint in a session. The codec are identified by a string name, which can be registered by any person or group with IANA (Internet Assigned Number Authority) [Han99].
- **Scalability:** H.323 was originally used to a single LAN. Issues like area addressing and user location were not concerned. The newest version defined the concept of a zone, and defines procedures for user location across zones for e-mail names. However, for large and complex location s, H.323 still has salability problem. SIP uses a loop detection method by checking the history of the message, which can be performed in a stateless manner [Sch98b]. Another difference is the SIP can run either TCP or UDP, while H.323 is TCP based signaling processing.
- **Service:** Roughly, SIP and H.323 provide the same services in the call control services. H.323 provides a much richer set of functionality. SIP provides more support for personal mobility services with the redirect and proxy server. But H.323 supports various conference control services that SIP does not provide.

The comparison and evaluate of both protocols are still undergoing. The question of who will survive or coexist is still not clear.

4.2.4 Summary

We have discussed a bunch of multimedia related protocols. In order to give a clear concept of the relationship between them, we use the Figure 4.7 presented in [Sch98]. The protocol types are distinguished by signaling protocol, QoS control protocol, and media transport protocol.

The protocols include SDP, SIP and H.323 for connection and signaling control. These signaling can be running both on TCP and UDP. The Quality of service protocol includes RTSP, RTCP, and RSVP. RTSP and RTCP are running over TCP or UDP. Because the RSVP will affect route protocol, it is also running over IP protocol. Only one media transport protocol, the RTP, is provided. The RTP is running over the UDP protocol. The name "real time protocol" does not mean it provide the real time transport service. It only gives the real time control information accompanied with RTCP. Based on RTP, the standards for carrying many codecs have been given in RFCs.

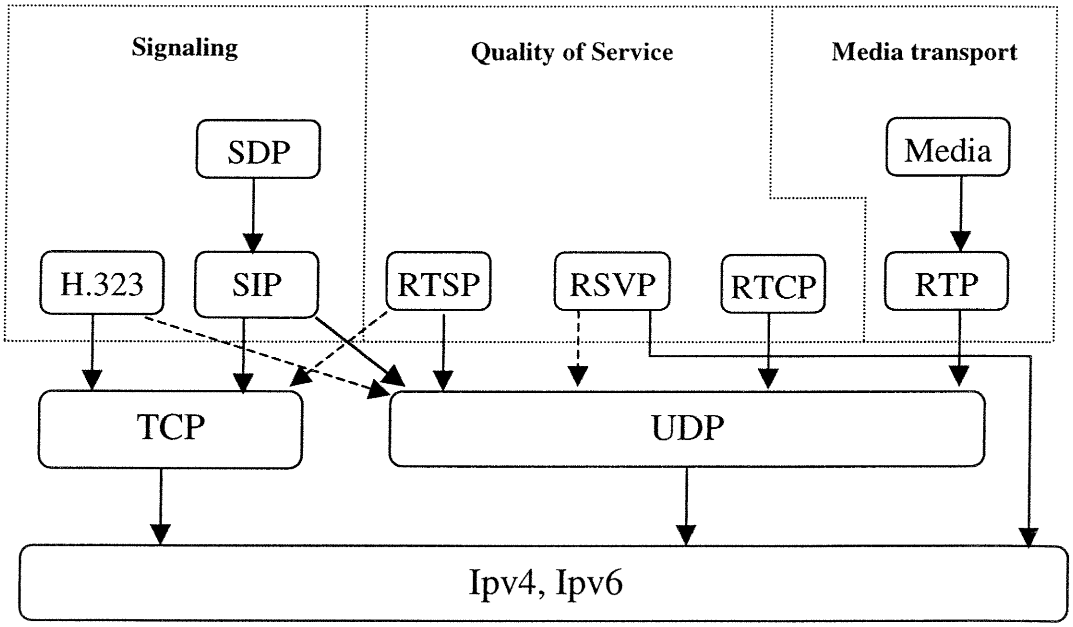


Figure 4.7: *Real-time related protocol stack*

4.3 Network resource management for multimedia applications

Network resource is very critical in the distributed multimedia application, especially in the time delivery sensitive application. I have introduced the IETF real-time protocol and related signaling and control protocols. But these protocols do not indicate how to control the network resource according to the application requirement during stream packet transmission. In this section, I will discuss how to control the network resource to satisfy application's requirement in more details.

4.3.1 Rate control mechanism

The requirement of multimedia application includes the data transmission throughput, delay, delay jitter and loss rate [Fer90]. Conventional packet switching data networks with window-based flow control [Jac88] and first-come-first-served service discipline can not provide services with strict performance guarantees. Thus, new rate-based flow control and rate-based service discipline have been proposed in the context of connection-oriented network architecture with explicit resource allocation and admission control policies [Zhh91].

Several rate-based scheduling disciplines have been proposed. Those algorithms for network traffic control include *Leaky Bucket* [Sid89], *Fair Queuing* [Dem89], Weighted Fair Queueing (WFQ), Worst-case Fair Weighted Fair Queueing (WF^2Q) [Ben96b], *Virtual Clock* [Zhl91], *Delay Earliest-Due-Date* [Fer90b], *Jitter Earliest-Due-Date* [Ver91], *Stop-and-Go* [Gol90], *Hierarchical Round Robin* [Kal90].

- **Leaky Bucket:** The *Leaky Bucket* algorithm described a simple model: each host at the network entrance puts packets from each data flow into a corresponding bucket that has fixed size. In other word, the bucket is a finite internal queue. The bucket opens per clock click to emit packets for transmission. When the bucket is full, incoming packets are discarded. This mechanism turns an uneven flow of packets from the user processes inside the host into an even flow of packets onto the network, smoothing out bursts and greatly reducing the chances of congestion. The Leaky Bucket is shown in Figure 4.8.

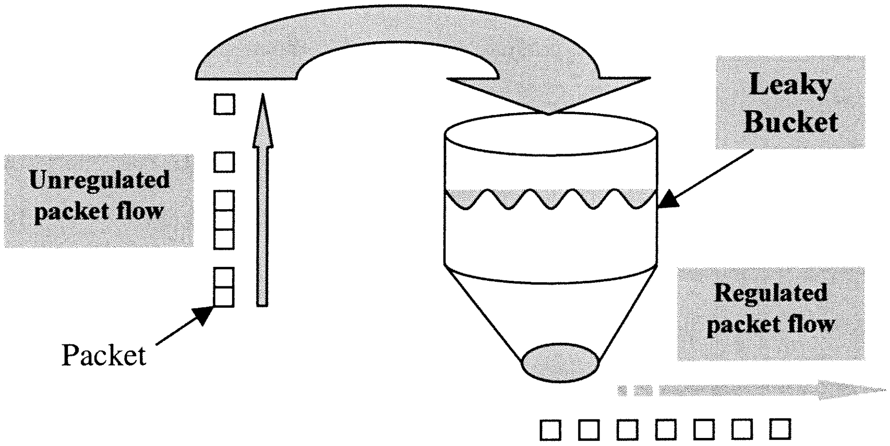


Figure 4.8: *Leaky Bucket*

- **Fair Queuing:** The aim of Fair Queuing is simple: if N channels share an output trunk, then each should get 1/N of the bandwidth, with the provision that if any channel uses less than its share, the slack is equally distributed among the rest. This can be achieved by doing a bit-by-bit round robin (BR) service among the channels. This is impractical, and so Fair Queuing tries to emulate BR. Each packet is given a finish number, which is the round number at which the packet would have received service, had the server been doing BR. By servicing packets in order of the finish numbers, it can be shown that Fair Queuing emulates BR. Channels can be given different fractions of the bandwidth by giving them weights; a weight corresponds to the number of bits of service the channel receives per round of BR service.
- **WFQ and WF^{2}Q :** Weighted Fair Queue add the weight to each queue which makes the host trade each queue not as same as the fair queue. The number of sent packets is based on the giving weight. For example, the host has six queues, Q1, Q2, Q3, Q4, Q5 and Q6. The weights of the three queues are 6,1,1,1,1,1. At each round, the host will send 6 packets from Q1, 1 packets from other queues. Same numbers of packets are sent in the next turn. The service order is given in Figure 4.9.

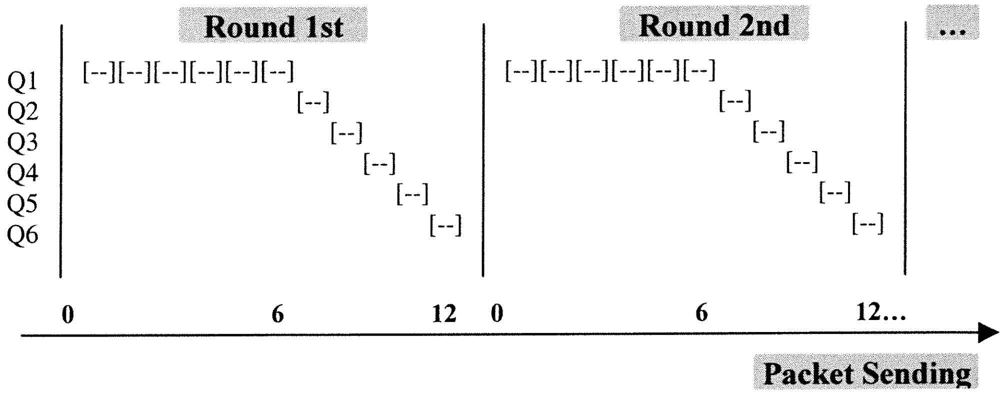


Figure 4.9: *WFQ Service Order*

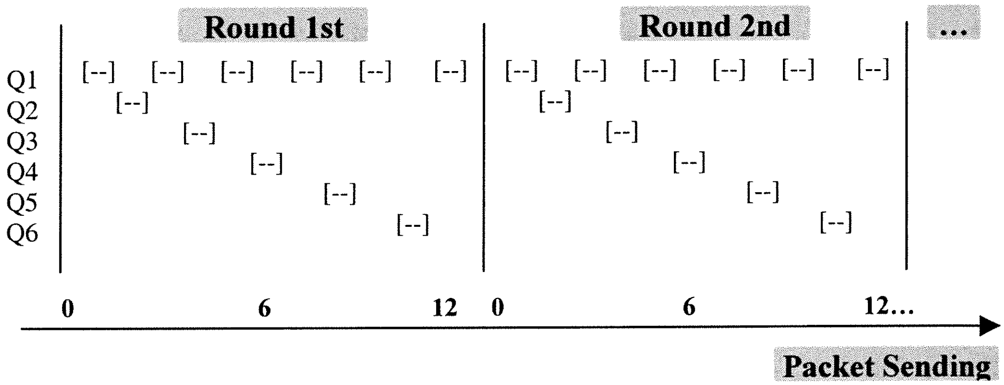


Figure 4.10: *WF^{2}Q Service Order*

WF^{2}Q is the alteration of WFQ. The major difference is it changes the queue service order. In Figure 4.9 and Figure 4.10 we can see the difference. The WF^{2}Q reduces the uneven of each queue send throughput. In order to implement it, WF^{2}Q use different clock algorithm for the packet sending [Ben96].

- **Virtual Clock:** The Virtual Clock discipline aims to emulate the Time Division Multiplexing (TDM) service discipline in the same way as Fair Queuing emulates BR. Each packet is allocated a virtual transmission time, which is the time at which the packet would have been transmitted was the server actually doing TDM. A simplified example: if a client is to get a service rate of 5 packets/second, incoming packets from that client are stamped with virtual service times 0.2 seconds apart. By sending packets in virtual time order, Virtual Clock can be shown to emulate TDM.

- **Delay Earliest-Due-Date:** In classic earliest-due-date (EDD) scheduling, each packet is assigned a deadline, and the packets are sent in order on increasing deadlines. The Delay-EDD service discipline is an extension where the server negotiates a service contract with each source. The contract states that if a source obeys a peak and average sending rate, then the server will provide a delay bound. The key lies in the assignment of deadlines to packets. The server sets a packet's deadline to the time at which it should be sent, if it had been received according to the contract. This is just the expected arrival time added to the delay bound at the server. For example, if a client assures that it will send packets every 0.2 seconds, and the delay bound at a server is 1 second, then the kth packet from the client will get a deadline of 0:2k+1. By reserving bandwidth at the peak rate, Delay-EDD can assure each channel a hard delay bound.
- **Jitter Earliest-Due-Date:** The Jitter-EDD discipline extends Delay-EDD to provide delay-jitter bounds (that is, a bound on the minimum as well on the maximum delay). After a packet has been served at each server, it is stamped with difference between its deadline and actual finishing time. A regulator at the entrance of the next switch holds the packet for this period before it is made eligible to be scheduled. This provides the required minimum and maximum delay guarantees.
- **Stop-and-Go:** The Stop-and-Go service discipline aims to preserve the `smoothness' property of traffic as it traverses the network. Time is divided into frames. In each frame time, only packets that arrived at the server in the previous frame time are sent. It can be shown that with this scheme, a packet receives both a minimum and a maximum delay as it goes from a source to a destination. Since the delay and delay-jitter bounds are linked to the length of the frame time, Stop-and-Go proposes multiple frame sizes.
- **Hierarchical Round Robin:** The Hierarchical Round Robin (HRR) server has several service levels, where each level provides round robin service to a fixed number of slots. A channel is allocated some number of service slots at a selected level, and the server cycles through the slots at each level. Figure 4.11 gives a relationship of each frame level. The time a server takes to service all the

slots at a level are called the frame time at that level. The key to HRR lies in its ability to give each level a constant share of the bandwidth. 'Higher' levels get more bandwidth than 'lower' levels, so the frame time at a higher level is smaller than the frame time at a lower level. Since a server always completes one round through its slots once every frame time, it can provide a maximum delay bound to the channels allocated to that level.

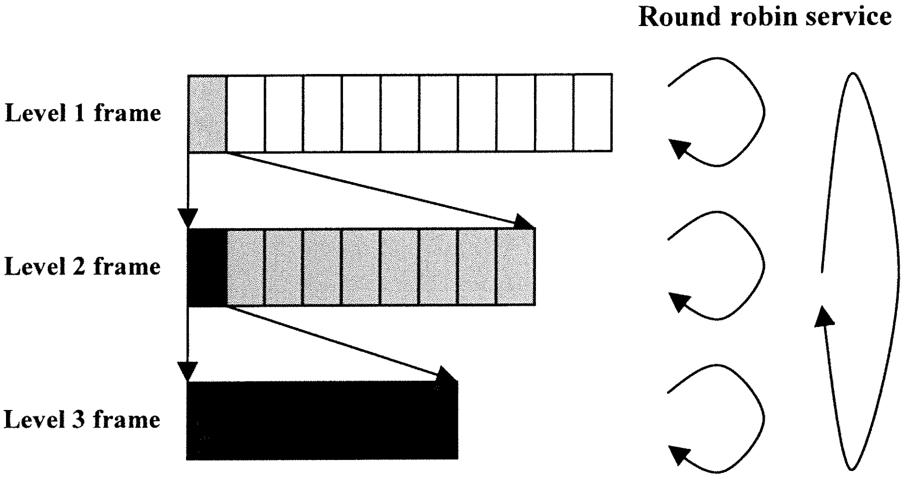


Figure 4.11: Hierarchical round robin frames

Many algorithms for network traffic control have been proposed. Each of them has different implementation proposal, different emphasis and is used under different circumstance. Also, based on these algorithms, lots of discussion for comparison has been published in [Zhh91] [Ben96] [Zhl91].

4.3.2 Error correction

The traditional mechanism for reliability provision is retransmission (e.g. TCP), which used an acknowledgment. If the acknowledgement is negative, the data are re-sent by the sender. But this mechanism is not very suitable to the multimedia application communication. The reasons are (1) with the traditional sliding window based flow control, the sender may be forced to suspend transmission while the continuous data flow is required; (2) the retransmitted data might be "out of data" when it reachs the client concerning the real-time characteristic; (3) It does not provide the multicast support.

To deal with the multimedia data losing problem, several strategies have been proposed:

- using UDP instead of TCP to reduce the acknowledgement time and prevent re-transmit "too late" data.
- using losing report (e.g. RTCP) to notify the sender to adjust the sending rate to reduce the losing rate.
- using the recently received packet to replace the lost data. This simple solution is useful to deal with the some kind of video packet losing. It includes the spatial and temporal interpolation [Bol98].
- using FEC (Generic Forward Error Correction) [Bol96c] to make the packet to carry redundant data to recover the recently missing data.

4.3.3 Dealing with *Delay* and *Jitter*

Figure 4.12 gives all possibility when delivering the UDP data. Because the none-guarantee UDP delivery characteristic, handling all problems listed above (Data Loss, Delay Jitter, Out-of-order, Duplication) belong to the high level protocols.

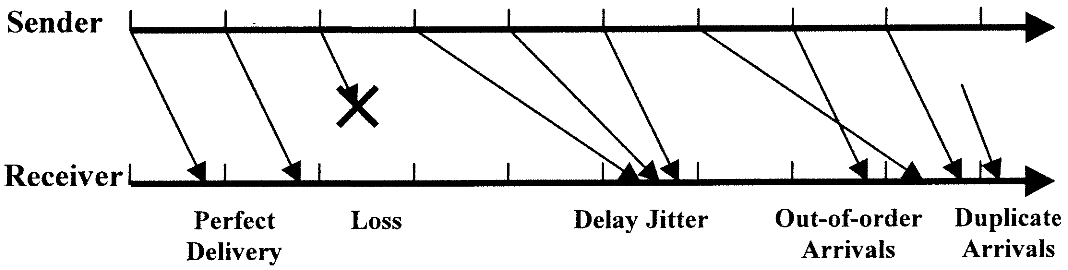


Figure 4.12: *UDP data delivery*

A simple way is using the delay-jitter buffer at the end system. When receiving RTP packet, the end system will not forward it to the codec construction layer for displaying, it will hold the packet for a little while to wait the "late coming" packet and reorder them. Then the packet will be delivered to the frame reconstruction module. This mechanism decreases the delay jitter and makes the packets smoothly transmitted to the frame reconstruction module. When it receives some packet coming too late to be useful, the receiver will discard it since the real-time multimedia characteristic. The system will treat this kind of packet as the "loss data". About how to deal the loss data and the loss data recovery algorithms are still under discussion in real-time related protocol. Several ways to deal this problem have been given in last section.

4.3.4 Resource monitoring

Figure 4.13 shows the famous sliding window mechanism for the TCP flow control proposed by Van Jacobson [Jac88]. It use the receiving acknowledgement number to control the sending data sliding window size, that is to make the data sending rate fit the network available bandwidth.

In RTP/RTCP based multimedia application, it can use the RTCP report to estimate the network resource status. According to the RTCP report, data sending rate control mechanism has been used. For example, when system find the received RTCP report which indicates the loss rate is higher than some threshold that affect the media quality seriously, it will decrease the packet sending rate to reduce the loss rate. If situation arises, new media codec should be provided for the lower network bandwidth.

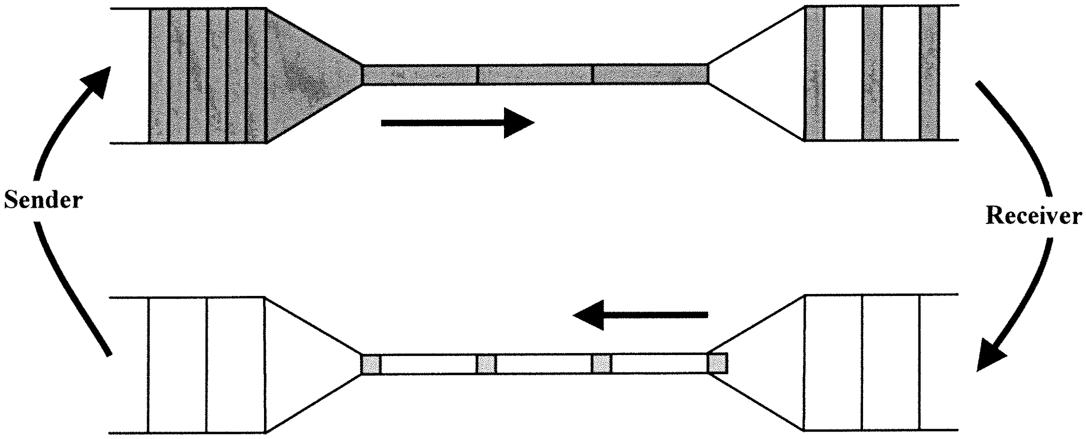


Figure 4.13: *Window Flow Control*

In application level, other network resource monitoring tools could also be used. One simple TCP/IP protocol application tool is "ping". "Ping" sends the ICMP message and listens to the ICMP echo message. It can provide the information like RTT (round trip time) and packet loss rate at the IP packet level. "TraceRoute" use similar mechanism but can provide more information. It can tell the RTT of each node along the path to the destination. In [Bol93], it uses the measured round trip delays of small UDP probe packets sent at regular time intervals to characterize the end-to-end packet delay and loss behavior in the Internet.

Another notion for bandwidth measuring is *packet-pair* [Kes91]. It uses two back-to-back packets that travel from source to destination and results in two acknowledgment packets returning. The calculation of bandwidth is based on the returning ACKs. The tools based on this theory have been implemented [Car96]. Figure 4.14 illustrates the journey of a pair of packets along the round-trip path from source to target and back. When the packets depart from the source host, the inter-departure gap is measured as (D2 - D1). After the packets go through the network, the inter-packet gap changed to "gap", which due to the network bandwidth limitation. When the packets return to the source the internal-arrive time can be measured by (A2 – A1), which equals to "gap". We can use the packet size and time got above to calculate the network available bandwidth.

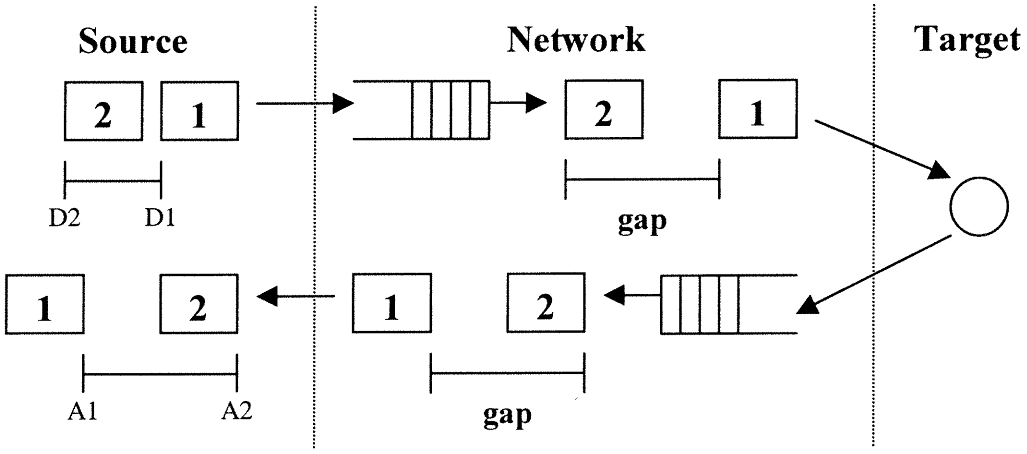


Figure 4.14: *Packet Pair*

There are several assumptions should be considered in the *packet-pair* methods: (1) there is no packet intervening between the two packets; (2) packets size should be big enough to occupy the network bandwidth in order to generate evident gap between the packets. (3) the probe packets should go along the path the same as the path the data transmitted.

4.3.5 Technologies used to improve QoS for VoIP

Voice over IP (also named Internet Telephone) is a very important multimedia application in current Internet. The marketing of VoIP is increasing very quickly. In order to provide better voice quality and utilize current IP network efficiently, several technologies are used and proved effective in the real application, which include:

(1) Echo Cancellation:

When a two-wire telephone cable connects to a four-wire PBX interface, a special electrical circuit called a hybrid is used to convert between two wires and four wires. Although hybrid circuits are very efficient in their conversion ability, a small percentage of telephony energy is not converted but instead id reflected back to the caller. This is called *Echo* [Micom].

If the caller is near the PBX, the echo comes back so quickly it can not be discerned. However, if the delay is more than about 10 milliseconds, the caller can hear an echo. To prevent this, gateway vendors include special function that listens for the echo signal. Echo network delay can easily by 40-50 milliseconds, so the echo from the far-end hybrid would be quite pronounced at the near end. Far-end echo cancellation eliminates this.

(2) Packet Prioritization

The router is instructed to look for voice packets and put them ahead of any data packets waiting in the router transmit queue. This way a string of out going data packets will not add to the variability of the arrival time of voice packets. Voice packet prioritization is especially important at the low WAN access speed.

The router is instructed to prioritize voice IP packets, either by the network administrator explicitly programming the router to look for the gateway's "well known UDP port number", or by using a prioritization protocol (i.e. RSVP).

(3) Packet segmentation

One important VoIP delay-management step is to ensure the very long data does not delay the voice packet from exiting router in a timely manner. This is achieved by programming the router to segment all outbound data packets according to the speed of the WAN access link.

The router is instructed to segment voice IP packets, either by the network administrator explicitly programming the router to segment all packets (data or voice), or by using RSVP.

(4) Silence suppressing

Silence detection technology recognizes the periods of silence in a conversation transmission, and stops sending IP speech packets during those periods.

A typical phone conversation, only about 30-40% of a full-duplex conversation is active. When one person talks, the other listens. And there are significant periods of silence during speaker's pauses between words and phrases.

But the silence suppression also brings some potential downsides. The first is "first word clipping". This occurs when the speaker begins talking, and the silence-suppressing technology does not recognize quickly enough and misses the first part of the word. The second problem can occur if the technology does not include a provision for background noise regeneration. Silence suppression renders the line absolutely silent to the listener, so much so that the line sounds dead. But, by inserting "comfort noise" or periodically sampling the true far-end background noise and regenerating it for the listener, the line sound active.

4.4 Summary

This chapter gave an overview of the current status of real time protocols and introduced some mechanisms for network resource management. It is not the intend of this chapter to give a complete, in fact almost impossible, introduction of all related techniques, the main purpose is to provide a brief background of our work, and the related reference that could be used for future extension work. In our implementation, it is impossible to support all of these protocols. Only part of them will be used. We notice many of these protocols and resource control mechanism have been use in many distributed multimedia applications.

If this chapter lays out some theoretical background, next chapter can be regarded as an application overview, in which we will cover some current multimedia applications. Some of them appeared in the academic research library and others are in the commercial field.

Chapter 5

Introduction of Current Multimedia Applications

In this chapter, we will give several multimedia application examples. This introduction does not try to list all popular distributed multimedia applications; instead only a few examples are given which have their own characteristic. The first four of these examples are commercial applications. The rest four applications are developed by university or research institute. The introduction of commercial applications gives a fast look and feel of how the Multimedia applications function in real world. The introduction on research projects is focus on how to solve the problems mentioned above. Actually, many successful ideas from research projects have been adopted in the commercial field.

5.1 Microsoft NetMeeting

Microsoft NetMeeting [NetMt] is a powerful tool that allows real-time communication and collaboration over the Internet or corporate Intranet. From a computer running Window 95, Windows 98, or Windows NT 4.0, users can communicate over a network with real-time voice and video technology. Users can also work together on virtually windows-based program, exchange or mark up graphics on an electronic whiteboard, transfer files, or use the text-based Chat program. It can be free downloaded from Microsoft Web site.

On the Internet, connecting to other NetMeeting users is made with the Microsoft Internet Locator Service (ILS), allowing participants to call each other from a dynamic directory within NetMeeting or from a Web page. NetMeeting uses the Lightweight Directory Access Protocol (LDAP) to support accessing the ILS server for dynamic directory information and facilitate point-to-point Internet communication sessions. LDAP is a standard method for program clients to query and access information stored on directory servers over TCP/IP connections. LDAP is derived from the X.500 global directory and the Directory Access Protocol (DAP).

At the media data transmission and call signaling level, NetMeeting adopts ITU H.323 protocols suite and support more codec standard other than H.323 supporting. Because NetMeeting is designed for corporate communication, many popular codec standards for audio, video, and data conferencing are supported.

With NetMeeting, people can communicate and collaborate with users of NetMeeting and other standards-based, compatible products. Users can support different kinds of network connection such as ISDN, LAN, or modem connection with different bandwidth.

Before the session, user should specify the network type in NetMeeting, like the user profile configuration, and the corresponding bandwidth value is used as the available throughput for the call. For example, NetMeeting uses the default setting of 435.19 Kbps over a LAN connection. According to the user's specification of the network type, system will select correspond codec algorithm and other related technologies to make the media data playing at good quality over the specific connections.

In a NetMeeting call, the highest priority is given to the audio stream, followed by the data stream, and then the video stream. During a call, the management system operates continuously to ensure smooth operation of NetMeeting. The bandwidth use of the audio stream is deducted from the available throughput. The data subsystem is queried for the current average size of its stream, and this value is also deducted from the available throughput. Then, the video subsystem uses the remaining throughput to create a stream of corresponding average size. If no throughput remains, the video subsystem will operate at a minimal rate and will compete with the data subsystem to transmit over the network. In this rare case, performance will degrade momentarily, as flow control mechanisms engage to decrease the transmission rate of the data subsystem. As a result, audio will sound clear, data conferencing will be functional, and video quality will be visually useful, even at low bit rates.

5.2 CU-SeeMe

CuseeMe [CucMe] is a video conference application through the Internet. It was originally developed by Cornell University and became commercial application later.

Features in CU-SeeMe include:

- Directory Service lets user see a list of all of the users published on a particular ILS server, whether they are using CU-SeeMe software or Microsoft NetMeeting;
- Conference Companion to locate associates, friends, or family online and call them without needing to know their IP addresses;
- View up to 12 video images simultaneously;
- Integrated T.120 data collaboration for sharing applications, whiteboard, and file transfer for multi-user collaboration during conferences;
- A choice of video and audio codecs for best performance over a variety of bit rates;
- H.323 protocol compatible.

5.3 Vocaltec Internet Telephone

Internet Telephone [VolCa] is an Internet Telephony software provided by Vocaltec corp. The software include client site and server site, and use the H.323 protocol as the signaling and controlling protocol worked with other IP based protocols. Both the client and server software is running on Windows NT and Windows 95 platform.

The server software includes (1) *VocalTec Gatekeeper* is the intelligent IP telephony service and control server that provides centralized addressing, security, and accounting; (2) *VocalTec Surf&Call Center Server* is A data collaboration server provides the interface between an incoming customer call, VocalTec Telephony Gateway terminals, and call center equipment; (3) *VovalTec Conference Server* is a software-only multi-conferencing unit for Internet and Internet-based conferencing.

The client software includes enhanced audio, live-motion video, and PC-to-phone calling. It enables users to simultaneously talk and see each other in real time for the cost of an Internet connection.

5.4 Real-video and Real-audio

Real System's Network Services [RealA] provides cross-platform methods for managing network communications. Any server-side or client-side Real system component can use Network Services to create TCP or UDP connections for reading and writing data. Network Services also provides interfaces that let components resolve DNS host names and listen for TCP connections on specified ports. Many broadcasting companies are using this Real-audio software to provide the real-time broadcast through Internet. Many famous broadcasting companies use this technology for on-line broadcasting including NBC, ABC, and CBC. With the development of Real-video software, some TV web sites also use it to join the real-time video broadcasting. The software can work on many platforms.

RM or RMFF (RealMedia File Format) is the media data format used in real-audio and real-video system. Third-party developers can convert their media formats into RMFF, enabling the RealMedia system to deliver the files to Real-Player or other applications build with the real-media SDK. Third-part developers can thereby use the RealMedia system to transport content over the Internet to their own applications.

5.5 NV

NV (Network Video conference) is one of the earliest Internet video conference tools and it is a video-only application developed by Xerox corp. One of major developer of this tool, Ron Frederick is one of the authors of the IETF RTP protocol.

The design of NV revolved around two major goals [Fre94]. One is to allow one to receive the video in standard window without requiring special hardware. Another is to make the program run over a wide range of network bandwidths. It should support sending low frame rate video over something as slow as a modem line, while also allowing higher quality video to be sent over the high-speed networks.

To achieve these goals, the system gives following steps to get the solution. First, it developed its own compression algorithm that makes the compression ratio reach 20:1. Second, it used the color, resolution and frame rate selection for the data rate control to make it adapt different network bandwidth.

The compression algorithm includes two-step process. First, the current frame is compared with the previous frame, looking for which portions of the image have changed significantly. Only those areas have changed will be retransmitted. We know this compression algorithm is called Inter-frame compression. It might reach the compression ratio of 3:1 or more. Second, based on the first step, each block is then compressed further using transform coding. This step provides compression ratio 6:1 or more. The transform coding include DCT and Haar wavwlet. The program will dynamically selects between them based on the system performance [Frederick94].

NV video streams use RTP/UDP as the transmission protocol. In order to deal with the data loss problem, the sender periodically re-transmit stationary blocks. This allows receivers who join a conference already in progress to get a complete image and also fills in any damage caused by packet losses after a short time.

At the time NV being developed, RTP version2 is still under definition. So the application has not utilized the whole advantage of the RTP/RTCP protocol, like the packet loss report.

The source code is available at ftp://ftp.parc.xerox.com/pub/net-search

5.6 INRIA Videoconferencing System (*IVS*)

IVS is a software system to transmit audio and video data over the Internet. It includes PCM and ADPCM audio codes, as well as ITU H.261 codec. Both Audio and video codecs are software codecs [Tur94].

IVS use UDP to transmit H.261 video data. In the H.261 standard, both CIF and QCIF, it breaks the data into a hierarchical structure that includes four layers, namely the picture layer, the GOB (Group of blocks), the MB (Macro blocks), and the Block layer. How to packetise them into the UDP/IP network is developed in *IVS* [Tur93]. Compared with TCP, UDP protocol brings quick speed and higher utilization rate of the network resource. But it has to face the data losing because of the characteristic of the UDP protocol. In *IVS*, it also addressed the scheme of how to implement packet loss recovery. Another future of *IVS* is it figured out how to monitor the network transmission quality and to make the flow control mechanism base on the information provided by the network monitor.

As the codec compression algorithm can be distinguished by Intra-frame and Inter-frame. The loss recovery strategies are different based on them. It is obviously that the Intra-frame will address more complex loss recovery algorithm. In Intra-frame, losing of a single packet might degrade video quality over a large number of frames, especially until the next Intra-coded frame is received [Bol98]. One approach to reduce the damage of the packet loss is to use simple loss concealment techniques like spatial and temporal interpolation as we mentioned before. Another approach applies to both inter- and Intra-coded frame is to use FEC-based (Forward error correction) error control mechanism. In this mechanism, the redundant information is transmitted along with the original information so that the original lost data can be recovered from the redundant information. It has the advantages of effective loss recovery without increasing latency and little additional bandwidth required. One FEC mechanism has already been implemented *IVS*. For more details about the implementation of FEC, please refer to [Bol98].

In order to utilize the all resource of the Internet and to provide good video quality as possible, *IVS* includes a feedback control mechanism [Bol94]. The parameters of the coder are adjusted according to the network condition observed. The output rate control is adjusted by changing either the video frame rate or the quantizer value and the movement detection threshold. The specific requirements of video application will indicate which of the three parameters should be modified when adjusting the output data rate. The frame rate is modified if the precise rendition of individual images is important. The quantizer and the movement detection threshold are changed if the perception of the movement is more important. The feedback in information consists in the loss rate calculated by the decoder and sent to coder. The packet loss rate is detected using the RTP packet sequence number [Bol94].

We know it is difficult to decide the frequency of feedback information sending. If it is too high, it will occupy too much the bandwidth, if too low, we can not get the accurate information of the network. This problem will increase with the group number of the participant increasing. In *IVS*, it adopts an approach that let the receiver send the NACK (negative acknowledgment) packet whenever it detects the loss of a packet, which decreases the feedback information. But when the number is larger then some number, say 10, another mechanism will be used to replace sending NACK packet. The replacement mechanism is the system will periodically send QoS report back to sender. Also to decide the period of sending the report is a problem. In its implementation, the receiver will send one QoS report after receiving every 100 packets. They also pointed out that ***each receiver sends feedback information at least once every 2 minutes*** [Bol94].

5.7 VIC

Vic is a video conferencing tool developed at UC Berkeley and LBL (Lawrence Berkeley Laboratory). It is another example for using the H.261 standard over the RTP protocol. *Vic* combined the advantages of *ivs* and *nv*, and provides more flexibility. The application support: (1) multiple network abstractions, (2) hardware based codec, (3) a conference coordination model, (4) an extensible user interface, and (5) diverse video compression algorithm [Mcc95].

It combined nv codec compression algorithm advantage with the H.261 standard to generate a new coding algorithm called Intra-H.261. It brings significant compression performance and improves in both run-time performance and packet-loss toleration compared with the previous two applications. It implemented the RTP payload specification for H.261 led to an improved scheme based on macro-block level fragmentation.

The V*ic* software architecture is built upon an event-driven model with Tcl/Tk interface. A set of objects is implemented in C++ and coordinated via Tcl/Tk. This combination provides the flexibility to cerate, delete and configure the C++ objects. The user interface provides control of bandwidth, frame rate, image size, encoding format by user's selection. Other services like multicast, synchronization separate audio and video are all supported.

5.8 RAT

RAT (Robust Audio Tool) is a multicast and unicast audio conferencing tool. It was developed from University College London. RAT can be used for both point-to-point videoconferencing involving a direct link between two hosts or for multiparty conferencing via Internet Mbone. RAT is based on IETF standards, using RTP above UDP/IP as its transport protocol, and conforming to the "RTP profile for audio and video conference with minimal control[Sch96b]."

The major features of RAT include [RAT99]:

- Development and implementation of speech coding algorithms as PCM, ADPCM, DVI, GSM and LPC.
- Half duplex mode or full duplex mode selection
- Packet loss repairing with redundant data
- Silence suppression for deducing data throughput
- Video synchronization
- Encryption support

The source code is available at http://www-mice.cs.ucl.ac.uk/multimedia/projects/rat

5.9 Summary

We have presented several distributed multimedia application examples in this chapter. These applications include both the famous projects in the research area and the popular commercial products. Each example has its own characteristic and the applying field. To provide a better understanding of these applications, I would like to give a comparison table to summarize their characteristics. It is impossible to present all attributes in a small table. We only give a brief comparison here. For more distributed multimedia application resources information, please refer [Vidco].

	Standard Support	Unique Futures	Network Support	Source Code	Platforms Support
MS Netmeeting	H.323	Collaboration	LAN/Internet/ISDN	N	NT/95
CU-SeeMe	H.323	Collaboration	Internet	N	NT/95/Mac
Vocaltec	H.323	IP Telephony	Internet	N	NT/95
Real-Media	Own Standard	File format	LAN/ISDN/Internet	N	Unix/NT/95
NV	RTP	First real-time video tool	Internet / Mbone	Y	Unix
IVS	H.261/FEC/RTP	Rate Control	Internet	Y	Unix
VIC	H.261/RTP	Flexibility	Internet / Mbone	Y	NT/95/Unix
RAT	RTP	Loss Repair	Internet / Mbone	Y	NT/95/Unix

***Table 5.1** A comparison among different multimedia applications*

From the table, we could find most commercial applications support the H.323 protocol. Actually, H.323 has been part of the industrial standard for multimedia application. We did not list the SIP supported application. In the IP telephony field, SIP has begun to be supported little by little. We also could find most source code of the applications from research library are free but these applications are not supported through many operating system platforms. Due to different research focus, each application of research field has its own unique future.

In the next chapter, we will present the implementation details of our own QoS management model. The implementation is not based on any available application listed above. But we refer some ideas from these applications. The focus of our implementation is on the QoS management.

Chapter 6
Implementation

6.1 Motivation

With the rapid increasing of Internet, many multimedia applications began to run over IP based network. Such kinds of applications include online-teaching, IP telephony, video conference, cooperate online-medicine application, online-shopping, real-time network game and so on. As we discussed before, current IP based network only provides "best effort" service. But the multimedia applications, especially the real-time applications, are usually sensitive to the network performance in terms of throughput and delay; therefore the current network QoS directly affects the multimedia data delivery. Those new constraints, or QoS Requirements, imposed by multimedia applications can not be handled gracefully by today's IP based network protocols.

Our proposal is to provide some kind of QoS management services that could serve the multimedia application to run over IP based network without suffering network resource shortage too much. We understand it is impossible to modify successfully and widely used Internet protocol suite. Therefore, our QoS management system for multimedia applications is designed upon the transport level, to make the application adaptable to the network resource changing and to provide the best presentation quality as it could. One point needs to be clarified here is that the QoS management system itself can not generate or reserve any network resource, it only tries to utilize the available dynamic network resource as much as possible. The strategies of how to utilize the resource are based on the user QoS profile given by each client, a user or an application program, before the media data transmission.

As a research project in university, it is easy to get the online-tutorial requirements than other multimedia applications. So our implementation of the project is based on the requirements of online-tutorial application.

Training is one of the most important applications for Internet technology, in general, and WWW in particular. A specific application involves a professor and many participants at different locations. A course session can be understood as several kinds of activities. The activities could be a collection of teaching activities that a professor is teaching using on-line slides, video and accessing database. The activities might include the participant involved activities like the participants are divided in several subgroups, each group working on a particular problem or issue. The activities also could be the participants' interaction with the professor using different possibilities that are associated with the capabilities of their system, either audio, video, mail, or white board.

The problems in this type of applications include: Quality of Service management, adaptation of the application to the QoS degradation, insurance of coherence and availability of data to all participants at the same time, management of collaborative work, hierarchical management of collaborative work, interactivity between professor and participants and between participants.

In the next section, the major implementation about the project coming from the concept we introduced previously is sketched. Originally, the project was oriented to the Online-teaching application, but many major modules of the implementation of the project are not limited in this field. Instead, they can be easily used to other IP based multimedia applications with minor modifications.

6.2 General framework of the prototype

Figure 6.1 illustrates the major implemented modules in our project and the relationships among them. The modules include QoS Manager, QoS Monitor, Streamer, Connection Manager, Transport Controller, Group Manager, User QoS Profile Manager, and User QoS Profile Input Dialog. We will explain the functionality of each part in the next paragraph. In Figure 6.1, the functionality about the QoS mapping, QoS admission control, and QoS adaptation are all included in the QoS manager module.

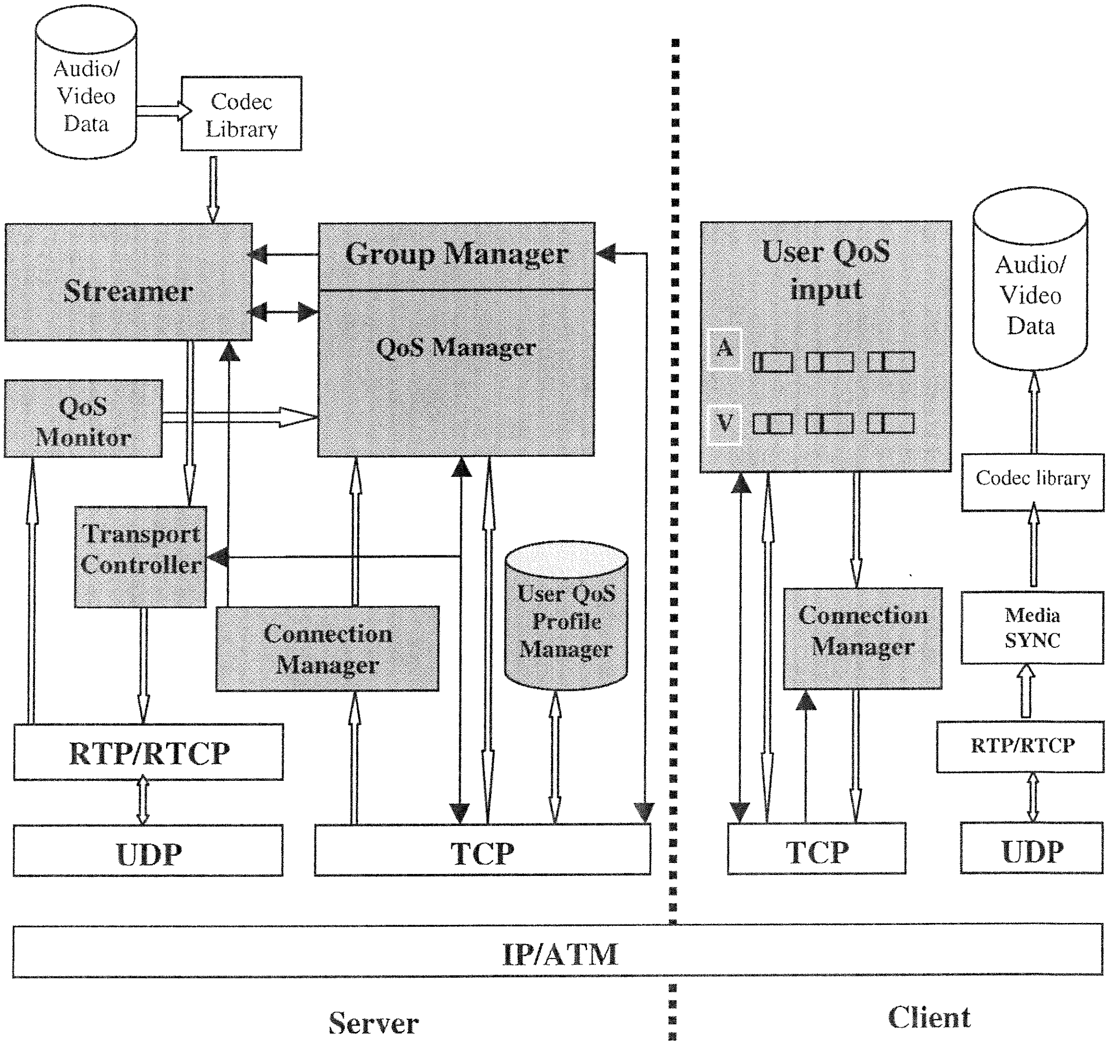


Figure 6.1: Big Picture of the project

One should notice that we use two different kinds of connecting arrows to describe the relationship between the modules, one arrow is narrow and the other is thick. In the graph, we use the narrow arrow to indicate the signal direction and the thick arrow to indicate the data transmission between modules. The signals include QoS admission control signal, Streamer parameter changing signal, Group admission signal, Transport Controller's network switch signal, and Connection Manager's control signal. The data transmission presented by the thick arrow includes QoS requirement, QoS Profile information, QoS Monitor report, and the video, audio data.

We will introduce the major tasks of each module following. For more details about the definition, please refer to the concepts introduced in previous chapters.

- **QoS Manager:** It is the key module of the system. Its components include QoS Mapping, QoS Adaptation, and Admission Control. The QoS request data and the QoS monitor information are sent back to the QoS Manager. Depending on the feedback information, QoS manager makes the decision how to react and send the response signal.
- **User QoS Input Dialog:** This module is in charge of the user QoS requirement acquisition. It provides several QoS levels and media priorities for selection. The user also can point out if the ATM network available for his/her application.
- **Streamer:** It is a simulator to generate the media data. Because the multimedia application uses many codec libraries to compress and de-compress data, and most of the libraries are not available directly, we use the Streamer to generate data to simulate different codec libraries. Because of the complexity of codec libraries, the integration of different libraries into our system will bring too much work, maybe more than QoS management system itself. We know that to develop and integrate the codec libraries is not the main target of our project, so we only use the streamer to simulate it. Whenever the codec libraries can be got and integrated easily, the replacement of Streamer becomes possible.

- **Group Manager**: Because our project is proposed for online-teaching, the group management is necessary. Different groups can be defined as the course name or seminar name. Each group is according to a specific course or seminar. For example, the group 'IFT6055' could be used for the course session whose code is 'IFT6055'. Each member in the same group should get same information at the same time, so the synchronization control be taken care of by the Group Manager.
- **Connection Manager**: Whenever a new session needs to be set up or a running session to be torn down, the communication source is changed. The Connection Manager is used to manage the communication ports, like UDP ports for RTP protocol transmiting data and the TCP ports for dilivery the control signal. When a new request arrives, the Connection Manager will check the communication port resource and try to get idle ports (include TCP and UDP port) for using. After checking this, it will respond the request to indicate whether the request is accepted or not. Connection Manager will release the communication resource when the session terminates.
- **Transport Controller**: It has several queues from the data sending scheduling and uses some kind of queuing theory for rate control. In our implementation, we use the WFQ (Weighted Fair Queue) to control the sending rate. We assign each queue a different weight according to the user QoS requirements and network monitoring information.
- **Codec Library:** Codec is used to compress the multimedia data into the digital packet data that can be stored and transferred. Different kinds of data have different codec formats. When we talk about the codec, lots of technologies are usually involved in, which beyond our research field. In our implementation, we use different Codec libraries to implement the switching between QoS levels. We could find hundreds of codec types and each uses different compression algorithm. In our predefined QoS level definition, we introduced H.263 for video, G.732 and Mp3 for audio. Both of them are well defined and widely used in many multimedia applications.

- **RTP/RTCP protocol stack:** In our implementation, we use RTP/RTCP protocol for media data transportation and data delivery quality statistic. The RTP/RTCP protocol library derives from the Elemedia Corp. [Eleme], which is part of their H.323 software development packages.
- **Media SYNC:** When audio, video and other related data arrive to the client, if all of them belong to same source, they should, ideally, are all played at the same time. Unfortunately, this is not always the case. Because the unpredictable characteristics of the IP based network, we don't know which type of stream packet will come first and when the other kinds of stream data belonging to same source will arrive. So the buffer control on the client site is needed to solve the different kinds of media synchronization problem. It is the major task for the Media SYNC module.

In Figure 6.1, we use two kinds of color to present the modules, gray and white. The gray parts indicate the modules what we were working on and have been implemented in our model. The white parts are the protocols and libraries that are already available in operating system or other systems, which can be used directly in our implementation.

This project is designed and implemented by a small group for the research purpose. The major parts that I designed and developed include QoS adaptation, Admission control, User group management, User QoS Profile management, User QoS input management and the QoS mapping. The rest of this chapter will focus on the discussion of these modules' functionalities. The core of these functionalities is ***QoS management***. All these modules I implemented provide sort of services for it.

6.3 Message exchange and signal sequence chart

In the implementation, we use three kinds of communication mechanisms. They are Windows Socket, Message Queue and Pipe.

Socket is the interface of TCP/IP network programming. The original Windows Socket comes from BSD UNIX and compatible with it (some of current Windows Socket functions have beyond BSD UNIX Socket definition). It provide two kinds of connection, stream based connection and datagram based connectionless connection, which is according to the TCP and UDP services provided in the TCP/IP protocol suite. We use both of them for media data and controlling signal transmission between client and server.

Another communication mechanism is Message Queue, which is provided by the Microsoft Windows system for message notification. We use Message Queue to exchange information and controlling signal between different modules in one system, both client and server site. In the server site, due to the speed limitation of the Message Queue when transferring large amount of data, we use Pipe technology for media data transmission between modules in one system.

In the following common use cases, the message sequence charts will give more clear idea on how the controlling messages being transferred. These use cases include *User Profile Access, Connection establishment, User Group Management* and *QoS Admission Control.*

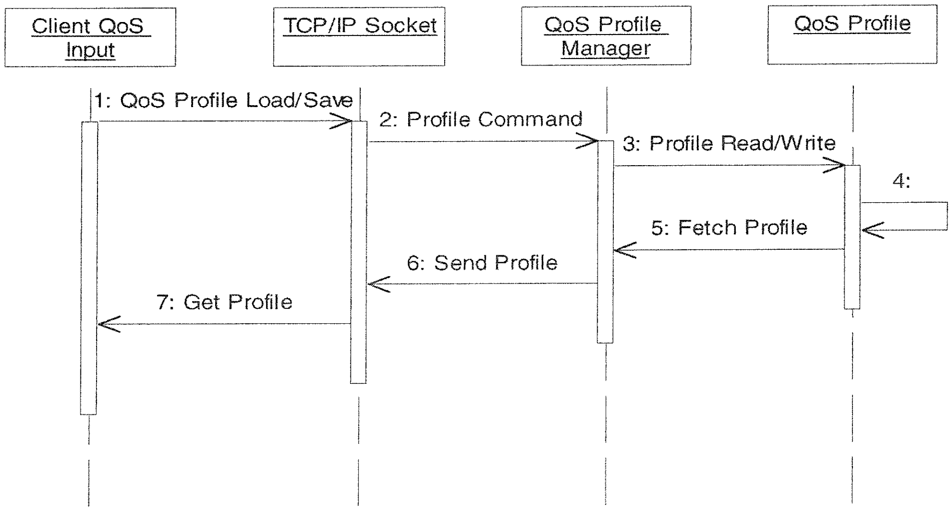


Figure 6.2: *MSQ of QoS Profile Access*

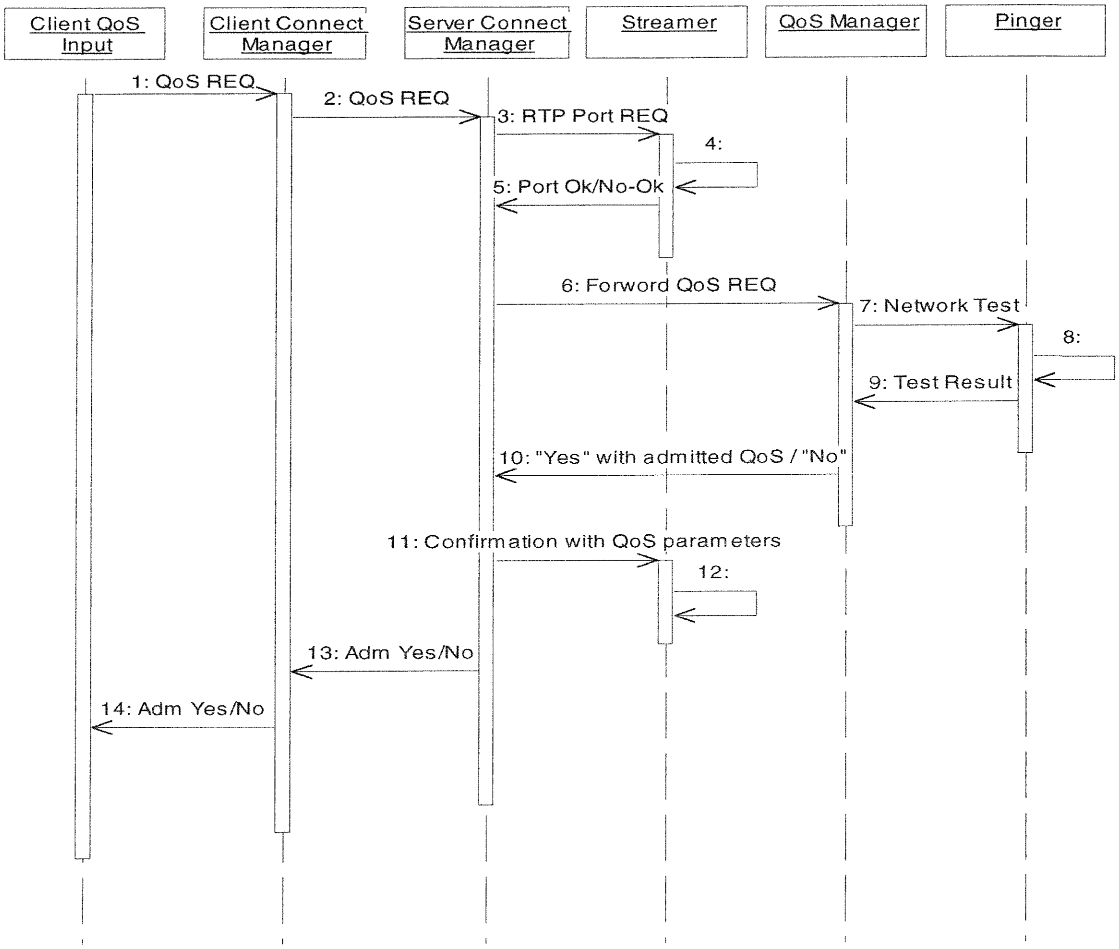


Figure 6.3: MSQ for Connection Establishment

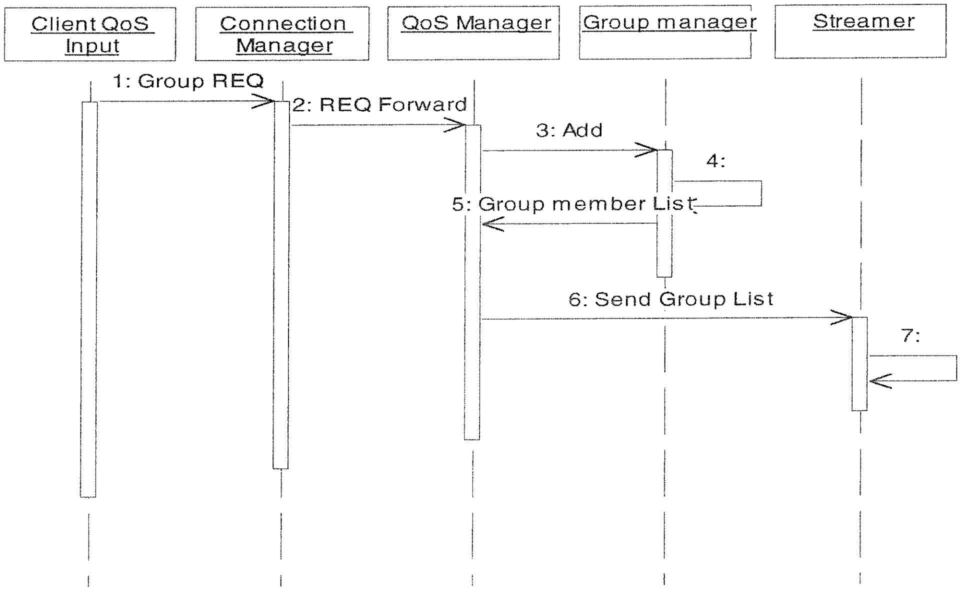


Figure 6.4: MSQ of Group Management

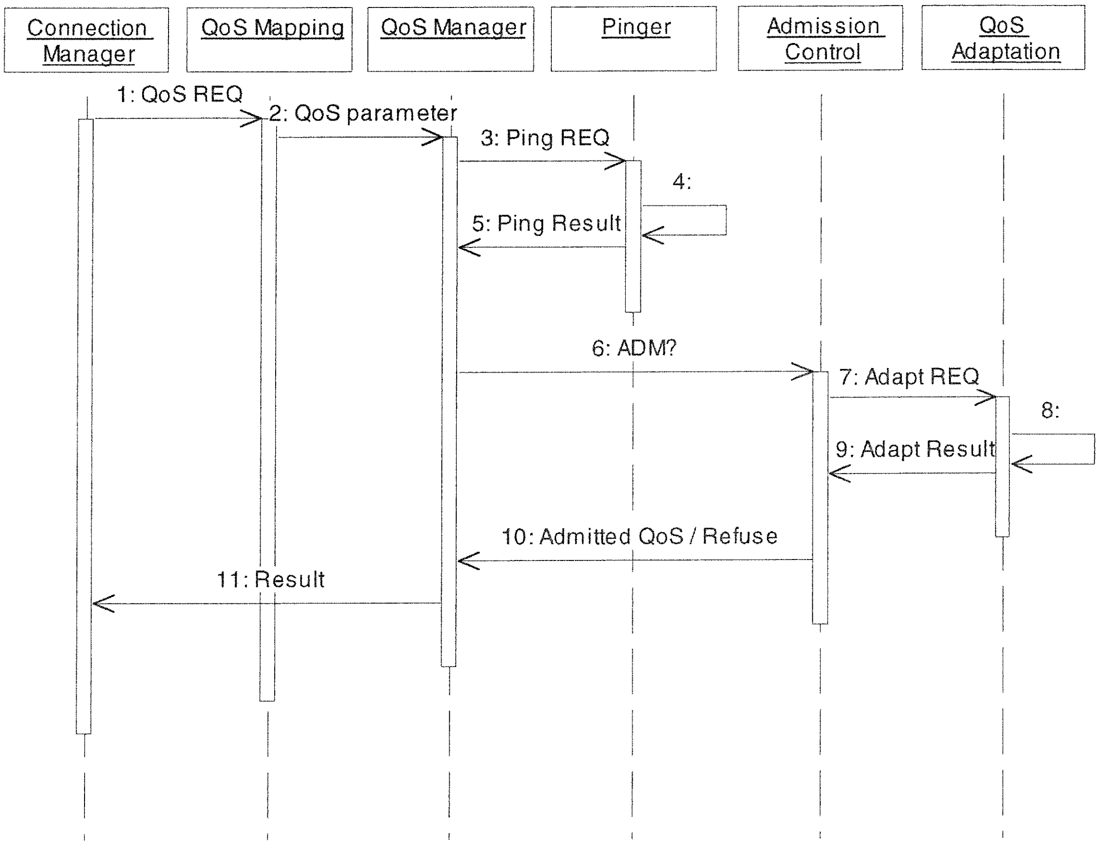


Figure 6.5: *MSQ of QoS Admission Control*

6.4 Stream data transmission and control

In Section 6.3, we have mentioned that we use the Windows Socket datagram service for transmission media data. After the Streamer generates the stream data to be sent to client, the data will be transmitted into a module named Transport Controller before being sent out through the Socket. The major functionality of the Transport Controller is to control the sending throughput based on the different media transmission throughput request. In the Transport Controller, there are several queues used for different media data. When a new packet comes, it will observe the header of the packet and send it to appropriate queue. Based on the media QoS request, the system assigns weights to different queues. For example, the audio codec G.711 has a throughput of 6.3 kbps and the video codec H.263 has a throughput of 56 kbps. The ratio of audio and video packet is around 1:8. Therefore, the Transport Controller will send 8 video packet and 1 audio packet in a round-robin fashion. This allows us to maintain the synchronization among media from same source.

6.5 QoS monitoring

QoS monitoring plays an important role in the QoS management. It provides all information the system needs for QoS violation detection, QoS Adaptation and Admission Control. In our system, we rely on two kinds of monitor mechanisms to get the network information. One is using the RTCP to get the feedback of the RTP data and another is using TCP/IP application "ping" to get the network information. The reason we need "ping" is before the RTP data being transmitted, we have no idea about the network situation. But we really need the reasonable information for the new session's admission control. So we use "ping" to collect the information of the network resource. Though this way is not an accurate calculation, at least we can get some rough idea of the network situation.

The Pinger is a thread running in our implementation in server site using the TCP/IP application "ping" to test the network. When a new session join request come, it will send a message to Pinger with the IP address. Then the Pinger will run a series of test to get the network throughput and delay, and then send it back the QoS manager for Admission Control judgement.

Another QoS monitoring report comes from the RTCP, which is a protocol collecting and reporting the information of the RTP data been transmitted. It resides on both client and server sites. After the client receives several RTP packets, it will send a RTCP feedback report to server. The report includes the statistic from last report. As we discussed before, the information in the RTCP report may contain delay, jitter, loss rate and so on, which is good enough for the QoS adaptation.

The monitoring report sending frequency is very important for the network information calculation. As other tuning factors, the proper value depends on how much trade-off you want to have. It is never easy to decide how often is proper. The two extremes of the setting have their pros and cons. If it is set too high, the network traffic will be jammed by the statistic report not the RTP data itself. If it keeps too low, it can not represent the network situation timely. In our implementation, the RTCP report will be sent after it getting 60 RTP packets initially. The RTCP report frequency can be adjusted easily if needed.

6.6 User QoS profile

This section will introduce our User QoS Profile management module. It will give the details of the QoS level definition and the QoS file format. At last, we use a figure to present the user QoS profile input interface.

6.6.1 User QoS level definition

There are many aspects to describe the multimedia application's quality from user's point of view, such as color, resolution, image size, image dot size, video frame rate, voice sampling rate, voice frequency range, and so on. It is also well understood that the quality of multimedia data also related to the codec library. If we would allow the user to choose any kind of media quality he expects, all the information mentioned before should be included. In a complete user QoS requirement profile, all these information is required. Unfortunately, it is impossible to give the user so much flexibility for QoS profile selection. One reason is that tuning all this parameters at a very fine level is impossible because that will depend on the codec implementation. Another reason is that many parameters are hard for user to understand and feel.

In order to make the QoS profile interface simple and easy to be implemented by system, as we mentioned before, we use the multi-level QoS profile for user's QoS selection.

We divide the Video QoS definition into five levels and Audio QoS definition into two levels. For each stream, the user can select the QoS preferring level, which is the user desired QoS, and the QoS adaptation level, which is used for the maximum tolerable QoS level for the QoS degradation. If the user does not indicate these levels, the system will give the default value. The QoS levels are defined in Table 6.1 and 6.2.

Level	Frame rate (/s)	Color (Bits)	Resolution	Compress Rate	Codec
1	30	16	CIF	27:1	H.263
2	30	16	QCIF	27:1	H.263
3	30	16	QCIF	54:1	H.263
4	10	16	QCIF	54:1	H.263
5	10	16	SQCIF	54:1	H.263

Note: *CIF: 352*288* *SQCIF: 128*96*
*QCIF: 176*144*

Table 6.1 *: Video QoS level definition*

Level	Sampling Rate (kHZ)	Quantization (Bits)	Codec
1	44.1	16	MP3
2	8	16	G.723.1

Table 6.2 *: Audio QoS level definition*

6.6.2 User Profile Format

We use one User Profile to record all users' profile. The file is composed by a list, which is composed by several separated units. Each unit includes one user's QoS selection. The information in one user profile unit includes user's ID, user desired

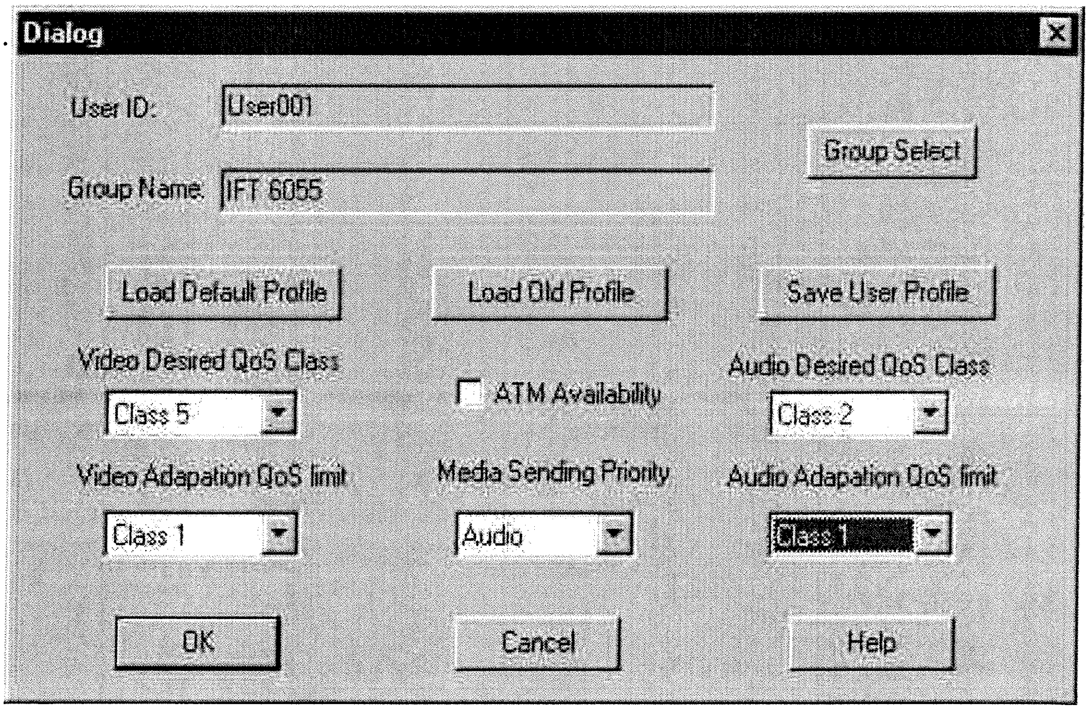
QoS level, the lowest user tolerable QoS level, media transmission priority, and ATM network availability. The details of the unit is described as follows:

User_ID	QoS_VD	QoS_VL	QoS_AD	QoS_AL	QoS_P

User_ID:	8 bytes;	User ID
QoS_VD:	2 bytes;	User Desired Video QoS Level
QoS_VL:	2 bytes;	Video QoS Level Degradation Level Limit
QoS_AD:	2 bytes;	User Desired Audio QoS Level
QoS_AL:	2 bytes;	Audio QoS Level Degradation Level Limit
QoS_P:	2 bytes;	QoS Guarantee Priority and ATM availability

6.6.3 Dialog presentation

Figure 6.6 is the user interface for user QoS profile input. In this dialog, user can select the desired video and audio QoS level and the QoS tolerable level. User can also save the QoS profile for future use or load the old profile filled out previously. The group information selection is also necessary in this dialog for the group management.



User ID: User001

Group Name: IFT 6055

Video Desired QoS Class: Class 5

□ ATM Availability

Audio Desired QoS Class: Class 2

Video Adapation QoS limit: Class 1

Media Sending Priority: Audio

Audio Adapation QoS limit: Class 1

Figure 6.6: User QoS Input

6.7 QoS Mapping

In Chapter 3, we have given the definition of QoS mapping. We will give more details about the implementation in the next paragraph. Before discussing our implementation, I will point out it is not easy to make the QoS Mapping work accurately in practice.

One reason is that different network layers have different packet fragmentation standard, which makes each layer's frame size different. It brings many difficulties for QoS mapping. For example, if one video frame size is 4k and one RTP packet size is 2k, when the RTP level packet loss rate is 5%, it does not mean the video frame loss rate is 5% too. Why? From the packet size given above, we get one video frame can be divided into two RTP packets. But losing two RTP packets does not equal to losing one video frame, since the two losing RTP packets may belong to different video frames. We can not predict whether the two losing RTP packets belong to same video frames or not because of the unpredictable nature of the IP network. Another reason is that the QoS mapping is closely related to Codec library and the Codec compressing algorithm makes the compression rate variable. For example, the video with more motivations has lower compression rate than the video with less motivations. It is hard to forecast too.

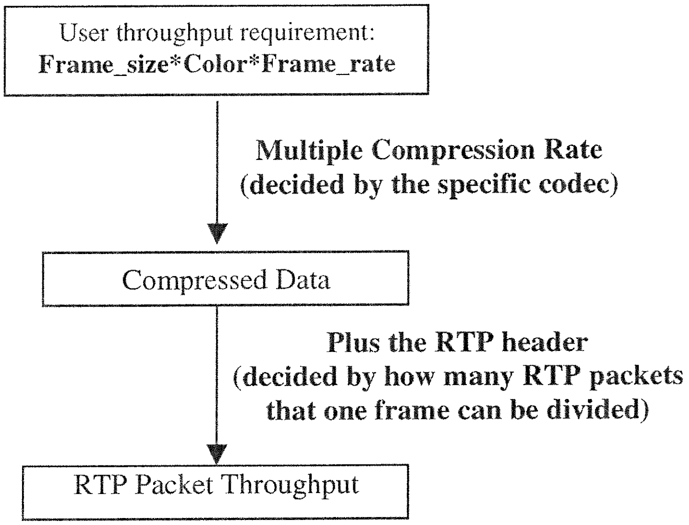
In our implementation, we made some assumptions to make the QoS Mapping work. One is that we assume all network layers user the same frame size, which means one media frame will be packed into one RTP packet. The second assumption is that the media compression rate is static. The third is that we do not consider the media frame header into throughput calculation because it also related to the Codec library closely.

6.7.1 Mapping calculation

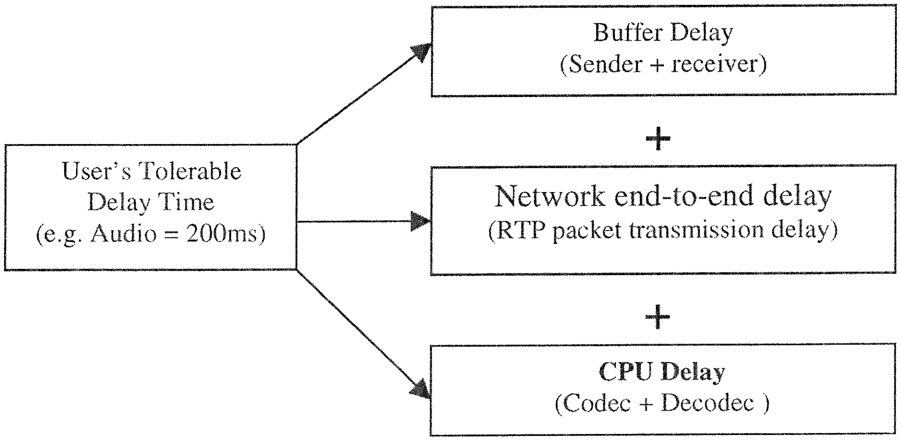
We use the following figures to illustrate the QoS Mapping calculation. Please note not all calculations are adopted in our implementation due to the complexity reason.

The QoS Mapping presented here are throughput mapping, network delay mapping, delay jitter mapping and the loss rate mapping. We use the figures below to present the QoS mapping calculation between different network levels.

(1) Throughput Mapping:


User throughput requirement:
Frame_size*Color*Frame_rate
Multiple Compression Rate
(decided by the specific codec)
Compressed Data
Plus the RTP header
(decided by how many RTP packets
that one frame can be divided)
RTP Packet Throughput

(2) Delay Mapping:

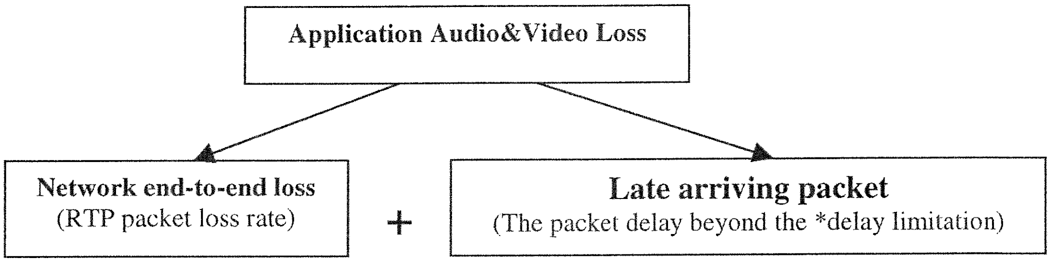


(3) Delay Jitter Mapping:

Because the ***Buffer Delay*** and the ***CPU Delay*** are constant for a given computer and codec algorithms, the delay jitter from the application layer is the same as the RTP transport layer:

User Delay Jitter Tolerate Time = RTP Packet Transmission Delay Jitter

(4) Loss Rate Mapping:


Application Audio&Video Loss
Network end-to-end loss
(RTP packet loss rate)
+
Late arriving packet
(The packet delay beyond the *delay limitation)

Note: **Delay limitation is such a period of time that holds all packets for a while in order to adapt the Delay Jitter.*

6.7.2 QoS Mapping Table

Table 6.3 and Table 6.4 are used for QoS Mapping in our implementation. These two tables give the threshold of each level definition. They include 5 video QoS levels and 2 audio QoS levels according to the QoS Profile definition.

QoS Level	Throughput (kbits/s)	Delay (ms)	Delay Jitter (ms)	Loss Rate
1	1200	150	50	5%
2	300	150	50	5%
3	150	150	50	5%
4	50	150	50	5%
5	27	150	50	5%

Table 6.3: *Video QoS Mapping Table*

QoS Level	Throughput (kbits/s)	Delay (ms)	Delay Jitter (ms)	Loss Rate
1	128	150	50	5%
2	5.3-6.3	150	50	5%

Table 6.4: *Audio QoS Mapping Table*

6.8 QoS adaptation strategy and Admission control

We have introduced the concept of adaptation strategy and Admission control in chapter 3. We will give more details of our implementations for these modules in this section. We will illustrate, implicitly, that these modules are the core functional parts in QoS management.

.

6.8.1 QoS adaptation strategy

QoS adaptation policies for both increasing QoS and decreasing QoS are all made from both the user QoS profile and QoS Monitoring information. But these two policies are used based on different situations. The QoS decreasing happens when QoS violation happens, but QoS increasing happens when QoS violation does not happen. The system tries to increase QoS when user application is always running on some "good" QoS monitoring report, but decrease QoS when the application can not function well because of the "Bad" QoS monitoring report.

Like QoS monitoring, the QoS adaptation itself will consume network and operating resource too. It is not worthwhile to increase QoS immediately when the application working well in current QoS situation. But it is necessary to decrease QoS immediately when application can not work well (QoS violation happens). So, the reaction speed for increasing QoS and decreasing QoS should not be same. We call this QoS adaptation strategy "***Slowly Increasing and Quickly Decreasing***"

What is this strategy talking about? For example, the QoS decreasing immediately when getting *one* "bad" QoS report that tells the QoS violation happens, but QoS increasing happens only when it receives *several* "good" report that tells current resource can satisfy the QoS requirement. In our implementation, the system tries to increase the media QoS parameters when it receives three "good" QoS monitor report continuously. The graphical comparison between these two situations will be presented in the next section.

Before explaining the details of the QoS Adaptation strategy, let us review the QoS profile definition first: (1) the user QoS profile defines the different media ***desired QoS levels*** and their ***last tolerable QoS levels*** for QoS level degradation; (2) the QoS profile also points out the media ***QoS guarantee priority***, this is which media QoS requirement will be satisfied with the limited resource. All concept defined in the QoS profile above will affect each step of the QoS Adaptation strategy.

The Adaptation Strategy will be divided into two groups. The first is to deal with the QoS decreasing and the second is used for QoS increasing.

(1) **Strategy for QoS decreasing** (QoS violation happens):

i. Check if the media transmission is on IP network, if yes, the degradation strategy will be performed on the IP network.

ii. Check the last QoS guarantee priority stream transmission level to see whether it has reached the last tolerable QoS level. If not, degrade one QoS level of current transmission and check whether the saved throughput by this action can satisfy the network shortage that causes the ***QoS violation.***

iii. If yes, the QoS degraded adaptation finished at this point where the network condition meets the user's degraded requirements. The modified QoS parameters (new QoS transmission level) will be sent back to the system for adjusting media transmission rate. If no, repeat step ii and step iii until the last QoS guarantee priority stream transmission level reach the last tolerable QoS level.

iv. The last QoS guarantee priority stream's transmission level degradation will be stopped at the last tolerable QoS level and the degrading of the higher priority stream will begin, which follows the same policy of step ii and step iii.

v. If all stream's QoS transmission level have been degraded and reached the last tolerate QoS level and the system still can not make it meet the decreased network resource, system will check the user profile to see if the user want network to be switched to the ATM network. If the user has agreed on transferring data on ATM network in the QoS Profile, all media transmission will be switched to ATM network. Otherwise, the system will stop the media transmission and notify the user that the QoS requirement can not be satisfied any more and ask what to do next.

(2) **Strategy for QoS increasing** (When getting three consecutive QoS monitoring reports and there is no QoS violation happen):

vi. Try to increase the QoS of the highest QoS guarantee priority stream to a higher QoS level if it has not reached the user's desired QoS level.

vii. If the highest priority stream has reached the user's desired QoS level or the system available resource is not enough to satisfy the highest priority stream to get a higher level, the same thing will be done to the next higher priority stream.

viii. If all streams meet the user's desired QoS level, the QoS increasing will stop there. There will be no further action even if some free network resource can be used.

ix. Clear the calculator for "good" QoS monitoring report.

After any strategy, both increasing and decreasing, having been processed, it should affect (1) some parameters used for the transport scheduling algorithms to control the different stream transmission, say the weight of the WFQ, and (2) media data generation parameter from the Streamer, such as frame rate, compression rate and so on. All these messages are transmitted in the server site using Message Queue.

6.8.2 Admission Control

The Admission Control is used for check whether a new session can be admitted or not. Actually, the admission control is used at two kinds of situations in our implementation, one is for the new session applying and another is for QoS adaptation.

We have introduced the major QoS parameters in Chapter 3, which include throughput, delay, delay jitter, and loss rate. But we don't consider all these parameters in the Admission Control. The major reason is Admission Control is used for future session, but most of the parameters except throughput are only used to describe the media data has been transmitted. Since it is hard to predict the IP based network behavior even the network throughput has been given. Because the other parameters may change when we issue a new admission, we only use the throughput for Admission Control judgement.

It should be pointed out that the parameters for Admission Control comparison are totally different from the QoS violation judgement. On the contrary, the QoS violation considers the delay, delay jitter, loss rate except network throughput. We also explained that it is not easy to accurately predict the media throughput requirement because of the codec algorithm. When the QoS monitor report indicates the transmission delay, delay jitter and the loss rate can all satisfy the QoS requirement, it means the media data has been successfully delivered to the client site even the network throughput is not as much as we expected before.

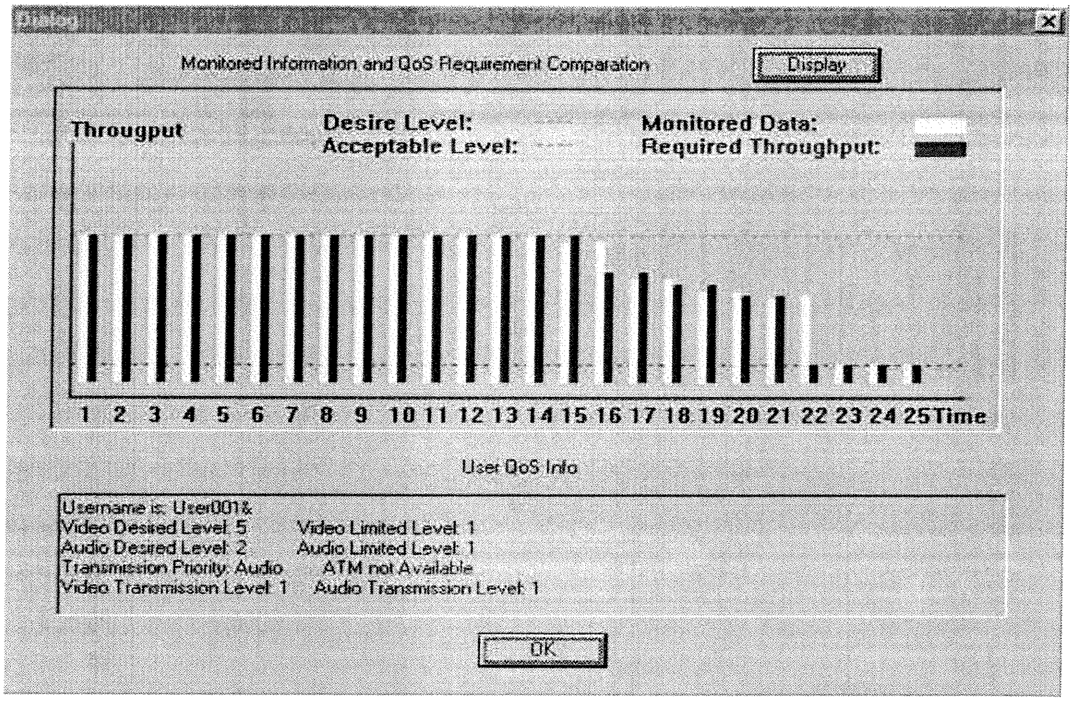
Table 6.5 gives the different QoS parameters considered for comparison between Admission Control and QoS violation judgement.

	Throughput	Delay	Jitter	Loss Rate
Admission Control	*			
QoS Violation Judgement		*	*	*

Table 6.5 *QoS parameters used for Admission control and QoS violation detection*

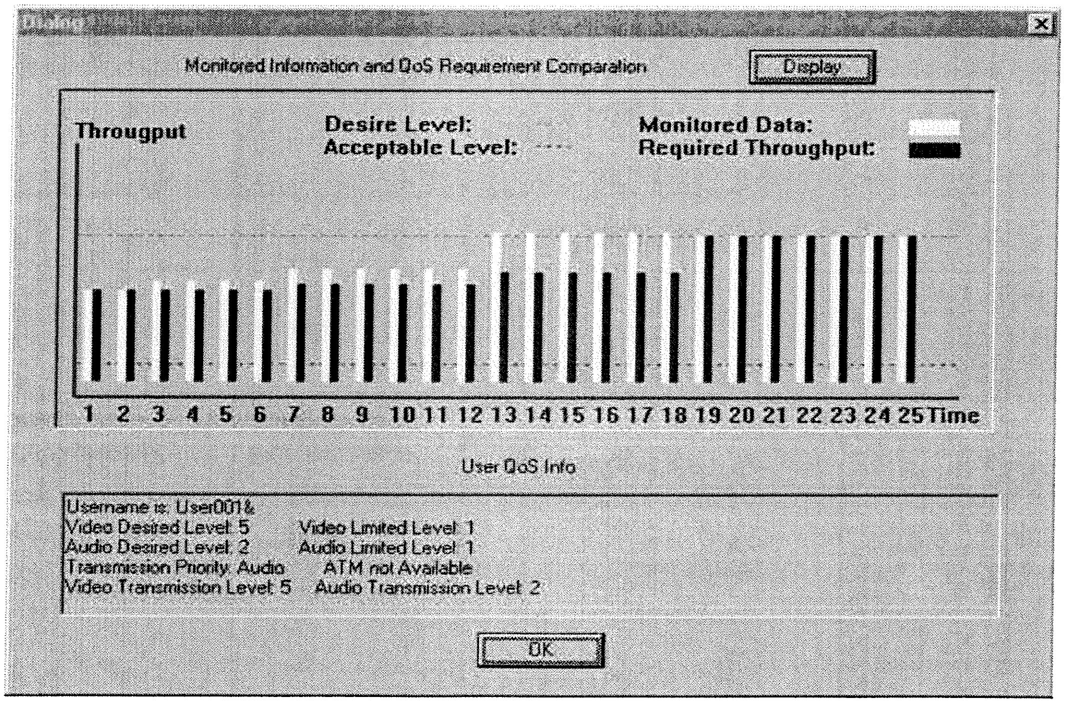
6.8.3 Graph presentation

We use two graphs to present the QoS adaptation result. Figure 6.7 is for QoS decreasing and Figure 6.8 is for QoS increasing. It can be seen from the figure that the reaction speed for QoS level increasing and QoS level decreasing is different. In the Figure 6.7, the QoS decreasing, at time slot 15, the monitored data (yellow bar) is less than the required data (blue bar), then the decreasing is reflected *immediately* on the *next* time slot, slot 16, as it is shown in the figure that yellow bar is higher than blue bar. In contrast, in Figure 6.8, QoS increasing, only when the indication of the yellow bar higher than blue bar lasts for several consecutive time slots, the QoS increasing occurs. This experiment results in consistency with the proposed differentiated adaptation strategies: “*Slowly Increasing and Quickly Decreasing*”.


Dialog
Monitored Information and QoS Requirement Comparation
Display
Througput
Desire Level:
Acceptable Level:
Monitored Data:
Required Throughput:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 Time
User QoS Info
OK

Username is: User001&
Video Desired Level: 5 Video Limited Level: 1
Audio Desired Level: 2 Audio Limited Level: 1
Transmission Priority: Audio ATM not Available
Video Transmission Level: 1 Audio Transmission Level: 1

Figure 6.7: *QoS Adaptation for Throughput Decreasing*


Dialog
Monitored Information and QoS Requirement Comparation
Display
Througput
Desire Level:
Acceptable Level:
Monitored Data:
Required Throughput:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 Time
User QoS Info
OK

Username is: User001&
Video Desired Level: 5 Video Limited Level: 1
Audio Desired Level: 2 Audio Limited Level: 1
Transmission Priority: Audio ATM not Available
Video Transmission Level: 5 Audio Transmission Level: 2

Figure 6.8: *QoS Adaptation for Throughput Increasing*

6.9 Summary

In this chapter, we presented the detailed policies used by the QoS management modules that involved in our project, in particular, QoS adaptation policy was proposed. We also illustrate the dialog GUI and simulation results of our implementation - the QoS management model for multimedia application.

Our model basically uses the RTCP feedback information to control the media data quality at the server site according to the user profile information. The original idea of this project was coming from the requirement of the online-teaching over Internet, but our implementation is not limited on such application. After merging with the upper level application, the model can be used for other distributed multimedia applications over RTP/UDP network for QoS management. Even we tried to make the model work properly, many aspects still need to be considered. We will list some major concerns we proposed for the future works in next chapter.

Chapter 7

Conclusion and Future Work

This thesis attempted to highlight some of the issues and concerns over the QoS management for Multimedia applications. In particular we select the ***QoS adaptation*** as our study focus. The discussions are based on the concept of distributed multimedia application, QoS related concepts, and real-time multimedia application protocols. For the multimedia delivery technologies over Internet to be in full swing, these issues must be addressed and agreed on. Some policies and strategies to solve the QoS adaptation gracefully have been proposed. In addition, a prototype implementation focusing on some key aspects – QoS management model for multimedia application over IP based network, was also introduced.

We observed that the research area of this field is still popular and not fully fledged, it seems that the new technologies and standards emerge with each passing day. With the development of the Internet, we believe that the multimedia applications over IP network will present a promising future. This is the main motivation of my topic.

As many popular and emerging technologies, the spectrum of research areas on QoS management for multimedia application is very wide. In this thesis, we only limit our work on a small extent of QoS and Multimedia area. But, it is either impossible to cover too many points especially when it comes to implementation. Thus, for our implementation, we focus on the model of QoS management control from the server site based on the RTCP feed back information.

The major contribution of our implementation is that it gives a general model of controlling the multimedia application QoS over the IP network, which can adapt to any applications running on it. By implementing this model, it also provides us a better understanding of QoS control for the whole procedure of multimedia application and the behavior of IP network.

My major contributions in this project are:

- QoS adaptation strategy design and implementation,
- User QoS profile definition and input interface implementation,
- QoS mapping and admission control design and implementation.

All this modules are designed and implemented in our own strategies and parts of them refer to the ideas of some existing applications.

Some of the limitations of our approach are due to the restrictions we imposed on, which include the available Codec library and network traffic simulation tools. Some limitations are also addressed in the future work we attempt to introduce below. Of course, due to the limited knowledge we have, there must be some unclear concepts introduced and wrong understanding presented, which may induce evident shortcoming in the implementation of our model.

Though we try to make our implementation be a general model that can adapt to as many applications as possible running on it, there are still lots of work to be done. I will point out a few aspects should be considered in the future work below.

Some aspects can be done in the future work for our model include:

- **Synchronization should be considered**

  In our model, we are luckily to see the synchronization problem can not only be controlled in the client site, but also in the server site. The advantage of the server control is it decreases the load of the client site to get more CUP time and memory for media data decompression.

However, in our implementation, we did not consider the *Synchronization* of different media streams. In the real world, it is a big problem. We can not imagine the audio of the movie played has no relationship with the video content. Consideration of the packet data loss behavior for the IP network and the unpredictable network characteristic increase the complexity of media synchronization. The challenge here is also includes how you can effectively capture the dynamics and make a graceful degradation.

- **Operating system resource management be considered**

  In our QoS adaptation strategy, we only consider the network information getting from the QoS monitor to make the QoS adaptation strategy; we did not consider the operating system resources. We know the multimedia compression and decompression consume much local system resource.

  Sometimes, even the network resource can satisfy application requirement, the operating system resources will still limit the media presentation quality. For this particular situation, the QoS adaptation should consider both the network resource and operating system resource as well. If possible, the operating system resource scheduling algorithm should be given to reserve the resource to guarantee the multimedia application running in the good quality.

  Though we know the local machine is easily controlled as compared to the wide-area network, the operating system resource management is also a very complicated issue. But to completely implement the QoS adaptation, monitoring information for the operating system resource should be provided from the QoS adaptation perspective.

- **QoS Monitoring information should be more accurate**

  To some degree, QoS monitoring plays an important role for the QoS management. And the core issue is how to collect accurate information. The inaccurate information introduced by monitor can be further accumulated to a big error when passing to other function module such as QoS adaptation.

  In out QoS monitoring module, we use the "Ping" and RTCP report to collect the network information and the media data delivery status. We notice using this way to collect the QoS monitoring is not accurate enough.

  First, using "Ping" command to estimate the network status before transmitting real data can not get the accurate information. There are several reasons: (1) "Ping" collect the information at the IP protocol level not at the stream level or RTP protocol level, but the QoS adaptation strategy is running based on the calculation of the RTP level. (2) We don't know whether the media data are transmitted follow the same path of the ICMP message travel path generated by "Ping".

  Second, the RTCP feedback information is based on the RTP packet. In other word, it only makes the calculating at the RTP level. All monitoring calculation such as Throughput, Delay, Jitter and Loss rate is on the RTP level. But our adaptation strategy calculation is on the application data level. In some cases, the difference between these two levels makes the calculations not accurate enough. This could lead to the QoS Adaptation cannot work the way it is supposed to be. Based on above issues, more efforts are worthy to put on finding the way of making the QoS monitoring collect more precise information in order that QoS adaptation and other affected QoS function can work more accurately.

References

[Alf96] Macro Alfano and Nikolaos Radouniklis, *A Cooperative Multimedia Environment with QoS Control: Architectural and Implementation Issues*, TR-96-040, ICSI, Berkeley, CA, Sept. 1996

[Ben96] J.C.R. Bennett and Hui Zhang, *Why WFQ is not Good Enough for Integrated Services Networks*, Proceeding of NOSSDAV'96, Apr. 1996

[Ben96b] J.C.R. Bennett and Hui Zhang, *WF^2Q: Worst-case Fair Weighted Fair Queueing*, Proceeding of IEEE INFOCOM'96, Mar. 1996

[Bla98] S. Blake et. All, *An Architecture for Differentiated Services*, RFC 2475, IETF, Dec. 1998

[Bla00] Uyless Black, *Voice Over IP*, Prentice Hall, 2000

[Boc97] Gregor v. Bochmann and Abdelhakim Hafid, *Some Principles for Quality of Service Management*, Distributed System Engineering Journal, Vol 4, No. 1, P.16-27, 1997

[Boc99] Gregor v. Bochmann and Zhen Yang, *Quality of Service Management for Teleteaching Application Using the MPEG-4/DMIF*, International Distributed Multimedia Systems and Telecommunication Services, 6th International Workshop, IDMS'99, pp.133-145, Toulouse, France, Oct. 1999

[Bol93] J.C. Bolot, *Characterizing End-to-End Packet Delay and Loss in the Internet*, Journal of High-speed networks, Vol.2, No.3, pp.305-323, Dec. 1993

[Bol94] J-C. Bolot and T. Turletti, *A Rate Control for packet Video in the Internet*, Proceeding of IEEE Infocom'94, Toronto, Canada, pp.1216-1223, 1994

[Bol96] J-C. Bolot and T. Turletti, *Adaptive Error Control for Packet Video in the Internet*, Proceeding ICIP'96, Lausanne Switzerland, Sept. 1996

[Bol96b] J-C. Bolot and A.V.Garcia, *Control Mechanisms for Packet Audio in the Internet*, Proceeding of IEEE Infocom'96, San Francisco, CA, pp.232-239, 1996

[Bol96c] J-C. Bolot and A.V.Garcia, *The Case for FEC-based Error Control for Packet Audio in the Internet*, ACM Multimedia Systems, 1996

[Bol98] J-C. Bolot and T. Turletti, *Experience with Control Mechanisms for Packet Video in the Internet*, ACM SIGCOMM Computer Communication Review, Vol 28, No.1, Jan. 1998

[Bou99] Ch. Bouras, A. Gkamas and Th. Tsiatsos, *A Web-based Distributed Environment to Support Teleteaching: Design and Implementation Issues*, 10th International

Workshop on Database and Expert Systems Application, Florence, Italy, pp.906-911, Sept.1999

[Bra94] R. Braden, D. Clark and S. Shenker, *Integrated Services in the Internet Architecture: an Overview*, RFC 1633, IETF, June 1994

[Brb97] B. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin, *Resource reSerVation Protocol (RSVP)*, RFC2205, IETF, Oct. 1997

[Brt97] T. Braun, *Internet Protocols for Multimedia Communications*, IEEE Multimedia, July-Sept. pp.85-90 and Oct. – Dec. pp.74-82, 1997

[Cai99] L.N. Cai, D. Chiu, M. McCutcheon, M.R. Ito and G.W. Neufeld, *Transport of MPEG-2 Video in a Routed IP Network, International Distributed Multimedia Systems and Telecommunication Services*, 6th International Workshop, IDMS'99, pp.59-73, Toulouse, France, Oct. 1999

[Cam94] Andrew Campbell, Geoff Coulson and David Hutchison, *A Quality of Service Architecture*, ACM Computer Communications Review, Vol 24, No. 2, pp.6-27, 1994

[Cag94] G. Carle, J. Schiller and C. Schmidt, *Support for High Performance Multipoint Multimedia Services*, LNCS 882, Multimedia Transport and Teleservices, International COST 237 Workshop, Vienna, Austria, Nov. 1994

[Car96] Robert L. Carter and Mark E. Crovella, *Measuring Bottleneck Link Speed in Packet-Switched Networks*, TR-96-006, Boston University Computer Science Department, March 15, 1996

[Cho95] H.S. Cho, M.R. Fry, A. Seneviratne and V. Witana, *Towards a Hybird Scheme for Application Adaptivity*, LNCS 1052, Teleservices and Multimedia Communications 2nd International COST 237 Workshop, Copenhagen, Denmark, pp.177-191, Nov. 1995

[Chu97] C. Zhu, *RTP Payload Format for H.263 Video Streams*, RFC 2190, IETF, Sept. 1997

[Cla92] David D. Clark, Scott Shenker and Lixia Zhang, *Supporting Real-time Application in an Integrated Services Packet Network: Architecture and Mechanism*, ACM Computer Communication Review, Vol 22, pp.14-26, SIGCOMM'92, Oct. 1992

[Coc92] Ron Cocchi, Deborah Estrin, Scott Shenker and Lixia Zhang, *A Study of Priority Pricing in Multiple Service Class Networks*, ACM Computer Communication Review, Vol 22, pp.123-130, SIGCOMM'92, Oct. 1992

[Dem89] A. Demers, S. Keshav and S. Shenker, *Analysis and Simulation of a Fair Queueing Algorithm*, Proceeding of ACM SIGCOMM 1989, pp.1-12

[Fer90] Domenico Ferrari, *Client Requirements for Real-Time Communication Services*, IEEE Communications Magazine, Vol. 28(11), pp. 65-72, Nov. 1990

[Fer90b] Domenico Ferrari and Dinesh Verma, *A Scheme for Real-time Channel Establishment in Wide-Area networks*, IEEE Journal on Selected Areas in Communications 8(3), pp.368-379, Apr. 1990

[Fer92] Domenico Ferrari, *Real-Time Communications in an Internet*, Journal of High Speed Networks, Vol. 1(1), pp.79-103, 1992

[Fer94] Domenico Ferrari, Anindo Banerjea and Hui Zhang, *Network Support for Multimedia: A Discussion of the Tenet Approach*, Computer Networks and ISDN Syatems, Vol. 26(10), pp. 1167-1180, July 1994

[Flo93] Sally Floyd and Van Jacobson, *Random Early Detection Gateways for Congestion Avoidance*, IEEE/ACM Transactions on Networking, Aug. 1993

[Flo95] Sally Floyd and Van Jacobson, *Link-sharing and Resource Management Models for Packet Networks*, IEEE/ACM Transactions on Networking, Vol 3 No. 4, Aug. 1995

[Flo97] Sally Floyd, Van Jacobson, Ching-Gung Liu, Steven McCanne and Lixia Zhang, *A Reliable Multimedia Framework for Light-weight Sessions and Applications Level Framing*, IEEE/ACM Transaction on Networking, Vol. 5, No. 6, pp.784-803, 1997

[Fre94] Ron Frederick, *Experience with real-time software video compression*, 6th International Workshop on Packet Video, Portland, Oregon, Sept. 26-27, 1994

[Gaf96] F. Garcia, D. Hucchison, A. Mauthe and N. Yeadon, *QoS Support for Distributed Multimedia Communications*, Proceedings of the 1st International Conference on Distributed Platform, Presden, Germany, Feb.-Mar. 1996

[Gol90] S. J. Golestani, *A Stop-and-Go Queuing Framework for Congestion Management*, Proceeding of ACM SIGCOMM'90, pp.8-18, Sept. 1990

[Gop94] R. Gopalakrishna and G.M. Parulkar, *Effect Quality of Service Support in Multimedia Computer Operating Systems*, Technical Report WUCS-TM-94-04, Department of Computer Science Department of Washington University, Aug. 1994

[Gop95] R. Gopalakrishna and G.M. Parulkar, *Real-time Upcalls: A Mechanism to Provide Real-time Processing Guarantees*, Technical Report WUCS-95-06, Department of Computer Science Department of Washington University, Sept. 1995

[Haf96] Abdelhakim Hafid, Gregor v. Bochmann, and Rachida Dssouli, *Distributed Multimedia Applications and Quality of Service*, Technical Report, TR-1036-96, University of Montreal, May 1996

[Haf96a] Abdelhakim Hafid, Gregor v. Bochmann, and Brigitte Kerheve, *A Quality of Service Negotiation Procedure for Distributed Multimedia presentational Applications*, IEEE Proceeding of 5th International Symposium On high Performance Distributed Computing, HPDC-5, Syracuse, New York, pp.330-339, Aug. 1996

[Haf98] Abdelhakim Hafid and Gregor v. Bochmann, *Quality of Service Adaptation in Distributed Multimedia Applications*, ACM Multimedia System Journal, Vol 6, Issue 5, pp.299-315, 1998

[Haf98b] Abdelhakim Hafid and Gregor v. Bochmann, *An Approach to Quality of Service Management in Distributed Multimedia Application: Design and an Implementation*, Multimedia Tools and Applications Journal, 1998

[Han99] M. Handely, H. Schulzinne, E. Schooler and J. Rosenberg, *SIP: Session Initiation Protocol*, IETF Internet Draft, Jan. 1999.

[Han99a] M. Handley, J. Crowcroft, C. Bormann and J. Ott, *Very Large Conferences on the Internet: the Internet Multimedia Conference Architecture*, Computer Networks 31(3), pp.191-204, 1999

[Hua96] J.-F. Huard, I. Inoue, A. A. Lazar and H. Yamanka, *Meeting QoS Guarantees by End-to-End Monitoring and Adaptation*, IEEE Proceeding of 5th International Symposium On high Performance Distributed Computing, HPDC-5, Syracuse, New York, Aug. 1996

[Hua97] J.-F. Huard and A. A. Lazar, *On End-to-End QoS Mapping*, IFIP 5th International Workshop on Quality of Service (IWQoS '97), New York, NY, May, 1997

[Jac88] Van Jacobson, *Congestion Avoidance and Control*, Proceeding of ACM SIGCOMM'88

[Jac98] V. Jacobson and M. Handley, *SDP: Session Description Protocol*, RFC 2327, IETF, Apr. 1998.

[Jon95] M.B. Jones, P.J. Leach, R.P. Draves and J.S. Barrera, *Modular Real-time Resource Management in the Rialto Operating Systems*, 5th Workshop on Hot Topics in Operating Systems (HotOS-V), May, 1995

[Kal90] C.R. Kalmanek and H. Kanakia, *Rate Controlled Services for Very High-Speed Networks*, IEEE Global Telecommunications Conference, San Diego, CA, Dec. 1990

[Kes91] Srinivasan Keshav, *A Control-Theoretic Approach to Flow Control*, Proceeding of ACM SIGCOMM, pp.3-15, 1991

[Kle98] Arno Klein, *Tele-teaching Scenarios for High Bandwidth Networks*, Computer Networks and ISDN System 30, pp. 1707-1716, 1998

[Kra98] Bruce Kravitz, *H.323 Technology*, White Paper of VTEL company, 1998

[Laz96] A.A. Lazar, K.S. Lim and F. Marconcini, *Realizing a Foundation for Programmability of ATM Networks with the Binding Architecture*, IEEE Journal of Selected Areas in Communications, Vol.14, No.7, pp. 1214-1227, Sept. 1996

[Mac95] S. MaCanne and V. Jacobson, *Vic: a Flexible Framework for Packet Video*, Proceeding of ACM Multimedia'95, San Francisco, CA, Nov. 1995

[Mat94] Laurent Mathy and Olivier Bonaventure, *QoS Negotiation for Muticast Communications*, LNCS 882, Multimedia Transport and Teleservices International Cost 237 workshop, Vienna, Austria, pp.199-218, Nov. 1994

[Nag93] R. Nagarajan, *Quality of Service Issues in High Speed Networks*, Ph.D Thesis, University of Massachusetts, 1993

[Pas98] J.C. Pasquale, G.C. Polyzos and G. Xylomenos, *The Multimedia Multicast Problem*, Multimedia System, Vol 6, pp.43-59, 1998

[Per97] C. Perkins, I. Kouvelas, O. Hardman, M. Handley, J.C. Bolot, A. Vega-Garcia and S. Frosse-Parisis, *RTP Payload for Redundant Audio Data*, RFC 2198, IETF, Sept. 1997.

[Pet00] L.L. Peterson and B.S. Davie, *Computer Networks A Systems Approach 2nd Edition*, Morgan Kaufmann Publishers, 2000

[Pla00] T. Plagemann, V. Geobel, P. Halvorsen and O. Anshus, *Operating System Support for Multimedia Systems*, Computer Communications V.23, pp.267-289, 2000

[RAT99] User Guide for RAT v3.0.33, University College London, Computer Science Department, 1999

[Ram96] Sanjeev Rampal, Douglas S. Reeves, and Ioannis Viniotis, *Dynamic Resource Allocation Based on Measured QoS*, 5th International Conference on Computer Communications and Networks, Rockville, Maryland, pp.24-27, Oct. 1996

[Rei99] Mark Reid, *Multimedia conferencing over ISDN and IP networks using ITU-T H-seties recommendations: architecture, control and co-ordination*, Computer Networks (31), 1999

[Ros99] J. Rosenberg and H. Schulzinne, *An RTP Payload Format for Generic Forward Error Correction*, IETF Internet Draft, Aug. 1999

[Sak94] Toru Sakatani, *Congestion Avoidance for Video over IP Networks*, LNCS 882, Multimedia Transport and Teleservices, International COST 237 Workshop, Vienna, Austria, pp.256-273, Nov.1994

[Sce99] Eric D. Scheirer, *Structured Audio and effects Processing in the MPEG-4 Multimedia Standard*, Multimedia System 7:11-22, 1999

[Sch96] H. Schulztinne, S. Casner, R. Frederick and V. Jacobson, *RTP: A Transport Protocol for Real-Time Application*, RFC 1889, IETF, Feb. 1996.

[Sch96b] H. Schulztinne, *RTP profile for audio and video conferences with minimal control*, RFC 1890, IETF, Jan. 1996.

[Sch98] H. Schulzrinne and J. Rosenberg, *Internet Telephony: Architecture and Protocols an IEFT Perspective*, Computer Networks and ISDN Systems, vol 31/3, pp. 237-255, Feb. 1999

[Sch98b] H. Schulzrinne and J. Rosenberg, *A Comparison of SIP and H.323 for Internet Telephony*, Network and Operating System Support for Digital Audio and Video, Cambridge, England, July, 1998

[Sch98c] H. Schulztinne, R. Lanphier and A. Rao, *Real Time Streaming Protocol (RTSP)*, RFC 2326, IEFT, Apr. 1998.

[Sch98d] H. Schulzrinne and J. Rosenberg, *An RTP Payload Format for User Multiplexing*, IETF Internet Draft, May 1998.

[She97] S. Shenker, C. Partridge and R. Guerin, *Specification of Guaranteed Quality of Service*, RFC 2212, IETF, Sept. 1997

[Shi99] Myung-Ki Shin and Jin-Ho Hahm, *Applying QoS Guaranteed Multicast Audio and Video to the Web*, IEEE Multimedia System'99, Florence, Italy, 1999

[Sid89] M. Sidi et. All, *Congestion Control through Input Rate Regulation*, IEEE Global Telecommunications Conference, Dec. 1989

[Ste95] R. Steinmetz and K. Nahrstedt, *Multimedia: Computing, Communications and Applications*, Prentice-Hall, 1995

[Tan96] A.S. Tanenbaum, *Computer Networks 3rd Edition*, Prentice-Hall, 1996

[Taw93] Wassim Tawbi, Linda Fedaoui and Eric Horlait, *Management of Multimedia Applications QoS on ATM networks Computer Networks*, Architecture and applications (c-13), IFIP, pp.15-26, 1993

[Tog99] J. Toga and J. Ott, *ITU-T Standardization Activities for Interactive Multimedia Communications on Packet-based Networks: H.323 and Related Recommendations*, Computer Networks 31, pp. 205-223, 1999

[Tow93] Don Towsley, *Providing Quality of Service in Packet Switched Networks*, Performance Evaluation of Computer and Communication Systems, pp.560-586, Springer Verlag, 1993

[Tur93] T. Turletti, *H.261 Software Codec for Videoconferencing over the Internet*, INRIA Research Report 1843, Jan. 1993

[Tur94] T. Turletti, *the INRIA Videoconferencing System (IVS)*, Connexions, Vol VIII, No. 10, Oct. 1994

[Tur96] T. Turletti and C. Huitema, *RTP Payload Format for H.261 Video Streams*, RFC 2032, IETF, Oct. 1996

[Ver91] D. Verma, H. Zhang and D. Ferrari, *Guaranteeing Delay Jitter Bounds in Packet Switching Networks*, Proceeding of Tricomm'91, Chapel Hill, NA, Apr. 1991

[Vog95] A. Vogel, B. Kerherve, G. Bohmann and J. Gecsei, *Quality of Service Management: A survey*, IEEE Journal of Multimedia Systems, Vol.2, No.2, pp.10-19, Summer 1995

[Wad96] Daniel Waddington, Geoff Coulson and David Hutchison, *Specifying QoS for Multimedia Communications within Distributed Programming Environments*, Proceeding of the International Cost 237 workshop, LNCS 1185, Barcelona, Spain, Nov. 1996

[Wad97] Daniel Waddington, Christopher Edwards and David Hutchison, *Resource Management for Distributed Multimedia Applications*, LNCS 1242, Proceedings of the 2nd European Conference on Multimedia Applications, Services and Techniques, ECMAST'97, Milan, Italy, pp95-121, May 1997

[Wit94] Hartmut Witting, Jorg Winckler, Jochen Sandvoss, *Network Layer Scaling: Congestion Control in Multimedia Communication with Heterogeneous Networks and Receivers*, LNCS 882, Multimedia Transport and Teleservices International Cost 237 workshop, Vienna, Austria, pp.274-293, Nov. 1994

[Xia99] X. Xiao and L.M. Ni, *Internet QoS: A Big Picture*, IEEE Network, Mar. /Apr., pp.8-18, 1999

[Zhh91] Hui Zhang and Srinivasan Keshav, *Comparison of Rate-Based Service Disciplines*, Proceeding of ACM SIGCOMM, 1991

[Zhh95] Hui Zhang, *Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks*, Proceeding of IEEE, Vol. 83(10), Oct. 1995

[Zhl91] Lixia Zhang, *VirtualClock: A New Traffic Control Algorithm for Packet-Switched Networks*, ACM Transaction on Computer Systems, Vol. 9, No. 2, pp. 101-124, May 1991

[CucMe] http://www.cuseeme.com

[Datab] http://www.databeam.com

[Eleme] Elemedia Corp, http://www.elemedia.com

[IETF] http://www.IETF.com

[IMTC] http://www.IMTC.com

[IVS] http://www-sop.inria.fr/rodeo/ivs.html

[Jef99] http://www.cs.unc.edu/~jeffay/courses/comp249f99

[Micom] http://www.micom.com

[NetMt] http://www.microsoft.com/netmeeting

[NV] ftp://ftp.parc.xerox.com/pub/net-search

[QoSforum] http://www.qosforum.com

[RAT] http://www-mice.cs.ucl.ac.uk/multimedia/projects/rat

[RealA] http://www.realaudio.com

[Vic] ftp://ftp.ee.lbl.gov/conference/vic

[Vidco] Videoconferencing Advisory Service, http://194.82.140.77/

[VolCa] http://www.vocaltec.com

[Xbind] http://www.ctr.columbia.edu/comet/xbind/xbind.html