Université de Montréal

An H.323-based Adaptive QoS Architecture

par

Jean-Marc Ng Wing Keng

Département d'informatique et de recherche opérationelle

Faculté des arts et des sciences

Mémoire présenté à la faculté des études supérieures

en vue de l'obtention du grade de

Maître ès sciences (M.Sc.)

en informatique

juin 2000

© Jean-Marc Ng Wing Keng, 2000

QA
76
U54
2000
V.040

Université de Montréal

Faculté des études supérieures

Ce mémoire de maîtrise intitulé

"An H.323-based Adaptive QoS Architecture"

présenté par

Jean-Marc Ng Wing Keng

a été évalué par un jury composé des personnes suivantes:

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

Mémoire accepté le: 26.09.2000

Abstract

The main characteristics of multimedia applications are the incorporation and manipulation of continuous media such as audio, video and animation. They require the transfer of delay sensitive information over a certain time period. One challenge facing these applications is to provide some guarantees so as to satisfy the requirements of users. The term Quality of Service (QoS) is the keyword that researchers use to address the set of requirements imposed by a user related to the performance to be achieved in the provision of services by an application.

In order to meet user desired QoS, QoS management is essential. It represents a set of activities working towards finding appropriate QoS characteristics of multimedia systems, to achieve their required functionality and optimize their performance. QoS adaptation is one of these activities, and makes applications more tolerant to inevitable fluctuations from the network. QoS adaptation is a complementary solution to reserving resources for providing some degree of service to applications. It is well suited for applications running on the Internet because the architecture of the Internet is founded on the concept that all flow-related state information should be stored in end-systems.

In this thesis, we propose a QoS architecture that allows the specification, formalization and mapping of user requirements from a high level to system-level and network-level parameters. It provides a framework for QoS control and management. We mainly focus on adaptation strategies that make a multimedia application gracefully cope with unexpected variations from the network. We address one particular aspect of adaptation, notably adaptation through multiple network configurations. We believe that the scenario comprising of a client and a server being connected via multiple networks is not farfetched.

We develop a prototype tool that considers on-line teaching and cooperative work as a case study, and which addresses the QoS issues mentioned in the preceding paragraphs. Our aim is to make the application developed comply by industry standards so that communication with other standards-based products is as seamless as possible. We achieve compatibility and interoperability by using the H.323 standard, which provides a foundation for media communications for packet-switched networks such as the Internet.

Keywords: Quality of Service, QoS adaptation, interoperability, H.323 standard, multiple network connections.

Résumé

Depuis leur conception, les systèmes de vidéoconférence sont restés dans le domaine des industriels et organisations multinationales à travers l'utilisation d'équipements coûteux, de lignes de communication dédiées, et de salles de conférence spécialement conçues. Ces systèmes, fonctionnant principalement sur des réseaux RNIS ou des réseaux locaux, ont vu un déploiement limité, créant une demande pour une implantation à plus grande échelle et moins dispendieuse.

Les récents bouleversements dans l'industrie des télécommunications, l'avènement des réseaux à large bande passante ainsi que de l'Internet, et de nouvelles techniques plus efficaces de compression d'audio et de vidéo, ont donné naissance à une multitude d'applications de vidéoconférence plus accessibles aux usagers. Ces applications génèrent des données multimédia distribuées sur des réseaux à commutation par paquets, comme les applications traditionnelles telles le courrier électronique et le transfert de données.

Les systèmes de télé-enseignement, basés sur le « Computer Supported Cooperative Work » (CSCW), connaissent une popularité fulgurante sur Internet et le Web. Le CSCW a été introduit pour désigner toutes les applications conçues pour des travaux de groupe. Un exemple implique un professeur donnant un cours à des étudiants, chacun situé dans un endroit différent. Le cours comprend plusieurs activités d'enseignement, comme l'acheminement en temps réel du cours à l'aide de transparents électroniques et de systèmes de stockage d'information, le travail collaboratif entre étudiants divisés en sous-groupes, et la participation des étudiants sur des applications partagées. De telles applications doivent fournir le support

nécessaire pour s'occuper des interactions d'usagers et de la gestion du travail de groupe.

Les applications multimédia ont le défi majeur de fournir une certaine garantie afin de satisfaire les besoins des usagers. Présentement, l'Internet ne peut offrir de telles garanties et offre un service « best effort ». La gestion du trafic de données est prise en charge de la façon la plus simple mais elle ne garantit ni la livraison ni la ponctualité. Cette provision de services convient aux applications telles le courrier électronique et le transfert de données, qui peuvent tolérer un délai de transfert raisonnable. Les applications multimédia par contre, sont sensibles aux délais, mais elles peuvent tolérer une certaine perte de données. Le terme Qualité de Service (QdeS) est le mot employé pour désigner un ensemble de caractéristiques, perceptibles par l'utilisateur et qui déterminent la performance du service qui lui est offert. Avec la prolifération d'applications sujettes à des besoins de QdeS, c'est un domaine sur lequel travaillent de nombreux chercheurs, mais sur lequel il reste beaucoup de progrès à faire.

Pour satisfaire les contraintes temporelles induites par la manipulation de données continues, un ensemble de fonctions de gestion de la QdeS est nécessaire. Beaucoup de recherche dans ce domaine s'est focalisé sur la réservation de ressources dans des réseaux de communication comme ATM. Ces réseaux offrent la possibilité de garantir à un certain point, la QdeS demandée par l'utilisateur. L'application de réservation de ressources pour Internet est un sujet de discussion car celui-ci favorise le stockage d'information sur les flux dans les machines clients. De nombreuses techniques basées sur la différentiation de paquets et de classes de services ont été proposées afin de fournir une certaine QdeS dans l'architecture d'Internet.

L'adaptation est une des activités de la gestion de QdeS qui permet aux applications d'être plus tolérantes envers les fluctuations de service du réseau dues à la congestion. L'adaptation de QdeS est complémentaire à la réservation de ressources, et contrairement à celle-ci, est applicable à des applications évoluant sur Internet. Ceci a donné droit à une pléiade de techniques d'adaptation conçues afin que les applications puissent réagir gracieusement à des variations inattendues du réseau.

Pendant les dernières années, la majorité des applications a été développée sur des technologies propriétaires, donnant droit à une gamme de produits pouvant communiquer entre eux mais pas avec ceux conçus par d'autres vendeurs. Il y a un consensus parmi les industriels d'aujourd'hui, de concevoir les applications basées sur les standards d'industrie pour permettre l'interopérabilité des applications. Cet appel à la standardisation s'est concrétisé en de nombreux ateliers offrant des environnements de vérification de compatibilité entre produits et conformité aux standards. Ces événements démontrent que bien que la volonté pour l'interopérabilité soit là, il reste du travail à faire en ce sens.

Buts et objectifs

Ce mémoire présente les travaux réalisés afin de relever quelques uns des défis mentionnés dans les paragraphes précédents. Notre but est de fournir un cadre pour la gestion de QdeS en développant une application adaptative aux fluctuations de QdeS et conforme aux standards.

Nous proposons une architecture qui gère la QdeS; de la spécification des besoins usagers à un haut niveau, à travers leur formalisation et traduction en des paramètres réseaux, jusqu'à la terminaison de connexion entre le serveur et le client. Nous incorporons plusieurs stratégies d'adaptation qui permettent à l'application de réagir face aux changements d'environnement, comme la congestion de réseau. Ces observations sont faites par une composante monitrice qui mesure la performance du réseau. Nous introduisons l'idée de réseaux multiples, où une machine client et une machine serveur sont reliées par plusieurs réseaux offrant une QdeS différente. Nous proposons un service au niveau transport qui gère ces multiples connexions réseaux, et qui fonctionne indépendemment du protocole de réseau sous-jacent.

Nous concevons l'application en tenant compte des besoins et fonctionalités du télé-enseignement. Un autre de nos buts est de faire en sorte que l'application soit conforme aux standards industriels courants, pour la compatibilité avec d'autres applications multimédia évoluant sur Internet.

Nous présentons les interfaces de l'application, de celui qui regroupe les spécifications de qualité de service et les paramètres de service réseau à l'interface de gestion de connexions à réseaux multiples. Des tests de performance par simulation sont appliqueés à notre application. Les résultats obtenus démontrent la validité de nos stratégies d'adaptation ainsi que nos choix de conception de l'architecture proposée.

Table of Contents

List of Figures

To

my grand parents,
my uncle and aunts,
and my brothers,

who are dearly missed.

Acknowledgements

First and foremost, I thank God for everything He gives me in life. Thank you my Lord for the realization of this work.

I cannot thank my parents enough for all the sacrifice that they have done for me. I owe everything to them. Thank you for always being there for your children. Thank you for being such wonderful parents.

I thank my sister Jennifer, my brother Jimmy, and Amanda very much for their unconditional support and encouragement.

I thank my director of research, Professor Rachida Dssouli, for all her availability, support, and advice, in and out of the academic field, during these 2 years.

I thank my friend Mohamed Salem for his advice and availability to always lend a helping hand. His contributions have been very instrumental, and very much appreciated.

I thank my colleagues and friends Mathieu Poirier and Yong Guo without whom this work would not have been impossible. Their advice and contributions have been very important and appreciated.

Je remercie M. Gérôme Canals pour ses conseils sur la publication de ce travail.

Je remercie le PCBF pour m'avoir donné la possibilité de poursuivre mes études de maîtrise à l'Université de Montréal.

Chapter 1

Introduction

1.1 Motivation

The practice of remote conferencing over the past two decades has been restrained by the use of expensive, dedicated equipment and lines, and specially-designed conference rooms, mostly confined to large organizations. Such circuit-switched conferencing was mainly based on ISDN (Integrated Services Digital Network) connections and has seen limited deployment, steering a demand for a less costly and more broad-based implementation of real-time multimedia communications.

Changes in the telecommunications industry, the advent of high-speed networks and of the Internet, faster processors, better data, audio and video compression schemes have given birth to a wide range of conferencing applications that are more accessible to the personal computing environment. Such Desktop Videoconferencing applications (DTVC) are being delivered on computer desktops over the same packed-based networks as the more traditional LAN applications like electronic mail and file-transfer.

On-line training [Robert *et al.* 97] is becoming one of the most popular applications on the Internet and on the World Wide Web (WWW) in particular. An example application involves a professor giving a course to students located in different locations. The course session can comprise of several teaching activities such as online course delivery through the use of slides and databases, collaborative working among students divided into various subgroups, participation and interaction among participants on shared applications etc. These applications need different types of tools like a videoconferencing tool for voice and audio communication, whiteboards and chat tools [Alfano 97]. Beyond these requirements, these applications need to provide support for the management of collaborative work including participant synchronization, conference control and integrity etc. [Reinema *et al.* 98].

One of the principle challenges facing on-line training and videoconferencing systems in general, is to provide some guarantees that the service requirements of the user will be satisfied. The current architecture of the Internet can only provide "best effort" service. Traffic is processed as quickly as possible but there is no guarantee in the timeliness and delivery [Zhao *et al.* 00]. This type of service provision is appropriate for applications such as electronic mail and file transfer, which can tolerate moderately long end-to-end delays. Multimedia applications, on the other hand, are diametrically opposite of fixed-content applications in terms of service requirements. They are more delay sensitive and loss tolerant [Schulzrinne 95]. The term Quality of Service (QoS) is the keyword used to represent the set of requirements imposed by a user related to the performance to be achieved in the provision of services by an application. With the proliferation of applications subject to QoS requirements, this is becoming and important area of research, in which progress still has to be made.

In order to meet the user service requirements, QoS management is essential [Bochmann & Hafid 96]. It is a set of activities working towards finding appropriate QoS characteristics of a multimedia system, to achieve its required functionality and optimize system performance. There has been much research conducted in the past years in the field of QoS management. Such work has mainly concentrated on resource reservation in communication networks such as ATM in order to provide some QoS guarantees to applications. The applicability of resource reservation for the Internet is a subject of much debate because its present architecture is founded on the concept that all flow-related state information should be in the end-systems. Other techniques, mainly based on packet differentiation and marking, are being proposed to provide a framework for QoS provision at the core level of the Internet.

One complementary solution to reserving resources is to make applications and other system components more tolerant to inevitable fluctuations in the performance in the supporting environment. This has led to the notion of adaptive applications that are designed to gracefully cope with unexpected variations from the network. QoS adaptation remains one of the prime activities of QoS management and is considered as one of the most pressing problems in multimedia research.

For the past years, the majority of such applications has been built upon proprietary technology and differing platforms, giving rise to an agglomeration of products that could just about connect among themselves, but not to other ones. Today, most industry professionals agree that the time has now come for those applications to be based upon industry standards so as to make the products compatible and inter-operable. This concern for standards compliance has recently taken epidemic proportions with the advent of industry-initiating and sometimes vendor-initiating events showcasing interoperability. Such events have portrayed the impression that although there is the will for multi-vendor cooperation, a lot of progress remains to be made to achieve full interoperability.

1.2 Goals and Objectives

In this thesis, we address some of the challenges mentioned in the preceding paragraphs. Our goal is to provide a framework for QoS management in realizing a prototype tool that is standards based and adaptable to QoS variations.

We propose an architecture that allows the specification of user requirements at a high level. These user parameters are formalized and mapped into system level and network parameters. We incorporate several adaptation strategies that react to changes in the environment resulting from traffic congestion. These observations are made by a monitoring module that measures the network performance. We introduce the idea of multiple network connections into our adaptation strategies. This concept is based on the fact that clients and service providers can be linked via more than one network. We propose a layer at the transport level to manage these multiple network connections.

We design our application taking tele-teaching as a case study and work in providing the functionalities for cooperative work. Our final aim is to make the application comply by industry standards so that communication and interoperability with other standards-based products would be as seamless as possible.

1.3 Thesis outline

In the next chapter, we discuss the need for interoperability and we present the H.323 standard, which provides a foundation for multimedia communications over networks that do not guarantee QoS. We give a detailed overview of the H.323 protocol, the components and protocols it supports; most notably, we describe the RTP protocol which is responsible for network feedback information in our architecture. Our architecture has been designed to be H.323 compliant.

In Chapter 3, we present the current techniques that are being used to provide QoS for applications running on the Internet. Emphasis is made on propositions for QoS provision at the core of the Internet and in end-systems. Several QoS architectures widely used in the literature are described and analysed.

Chapter 4 describes the principles of QoS management, starting from the activity of specifying user requirements to connection termination. We focus mainly on the activity of QoS adaptation which deals with variations in QoS provided by the network. We present several ways in which adaptation is implemented into applications.

In Chapter 5, we give a detailed description of our proposed architecture. We present the modules and components of the system and their interactions. We describe the design issues and the assumptions made. Our adaptation strategies and the management of multiple network connections are explained in finer details.

In Chapter 6, we describe the implementation phase in building the prototype tool. We present the interfaces used and describe the simulations conducted. Results in the form of graphs are included to demonstrate the viability of our architectural design and adaptation strategies.

Finally, we present some concluding remarks in Chapter 6 and discuss some future research directions for our work, including enhancements that can be made in the implementation of the proposed architecture.

Chapter 2

The H.323 standard

2.1 Need for Interoperability

As more and more multimedia products become available from a variety of multiple vendors, there is an increasing need for these products to be able to communicate properly with each other. Indeed, most industry experts, vendors and developers agree that the time for proprietary systems has passed, and that there is a need to build standards-based and fully interoperable multimedia products [Tehrani 00]. Standards ensure the following:

- Applications can work across different platforms and networks. It is commonplace for users in different locations running applications on varying platforms and networks, to communicate. Applications must be able to work beyond national boundaries.
- Applications with different features and functionality can still function together.
- Reduction in the time and costs in the product development cycle. It ensures that the technology is accessible to all interested parties and development can proceed without participants having to assume prior knowledge of other systems.
- Customers benefit from the competition among vendors in terms of product pricing and quality. They also gain from product upgrades based on their own requirements.
- Customers are more receptive to standard-based technologies rather than proprietary ones.

Two standards have emerged in order to provide interoperability among applications running on packet-based networks. One is the H.323 standard recommendation from the International Telecommunications Union (ITU-T) [url ITU-T], and the other is the Session Initiation Protocol (SIP) from the Internet Engineering Task Force (IETF) [url IETF]. Both protocols represent different approaches to the same problem: H.323 uses the more traditional circuit-switched approach while SIP favors the more lightweight Internet approach. At the outset of this thesis, there was a considerable wave of interest from industry professionals in favor of the H.323 recommendation. We hence decided to base our work on the H.323 standard.

2.2 Defining the H.323 standard

The H.323 standard provides a foundation for audio, video and data communications across networks, such as the Internet, that do not provide a guaranteed quality of service (QoS). It is part of a family of ITU recommendations that specifies the components, protocols and procedures for providing multimedia communication services over such networks [ITU-T H.323 98]. The current version, H.323 Version 2, has been approved in January 1998.

By complying to H.323, multimedia products from different vendors can inter-operate, and allow users to communicate without concern for hardware compatibility.

2.3 H.323 Components

H.323 defines four major components for a network-based communications system namely: terminals, gateways, gatekeepers and multiple control units (MCU). An example of an H.323 network comprising of these components is shown in Figure 2.1.

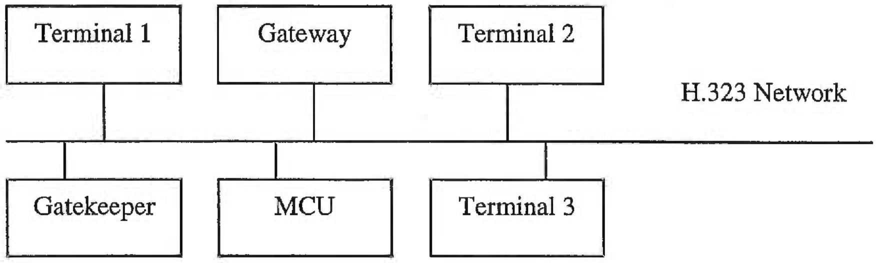


Figure 2.1: H.323 network overview

Terminals are client endpoints such as a stand-alone device or a personal computer (PC), that are used for real-time bi-directional multimedia communications. A gateway is an optional element in a H.323 conference. It provides many services, the most common being a translation function between H.323 conferencing endpoints and other terminal types. A gatekeeper acts as a central point for all calls within its zone and provides call control services to registered endpoints. It acts as a virtual switch. Finally, an MCU is an endpoint on the network, which provides the capability for three or more terminals or gateways to participate in a multi-point conference.

2.3.1 Terminal components

H.323 terminals are the client endpoints on the network that provide real-time, two-way communications. Figure 2.2 shows the components that the H.323 terminal consists of.

Audio codecs

The H.323 standard specifies that all H.323 compliant terminals must support voice communications. An audio codec encodes the audio signal from an audio input/output equipment such as a microphone for transmission on the transmitting H.323 terminal, and decodes the received audio code sent to the speaker on the receiving H.323 terminal.

The G.711 codec is a mandatory codec of the H.323 specifications. It transmits voice at either 56 or 64 kbps and is well suited for high bandwidth environments [Kostas *et al.* 98]. Other optional codecs such as G.723 (5.3 and 6.3 kbps), G.728 (16 kbps), and G.729 (8 kbps) are more suitable for lower bandwidth networks. G.723, in particular, is the default audio codec for many applications such as Microsoft NetMeeting [url NetMeeting] running on the Internet.

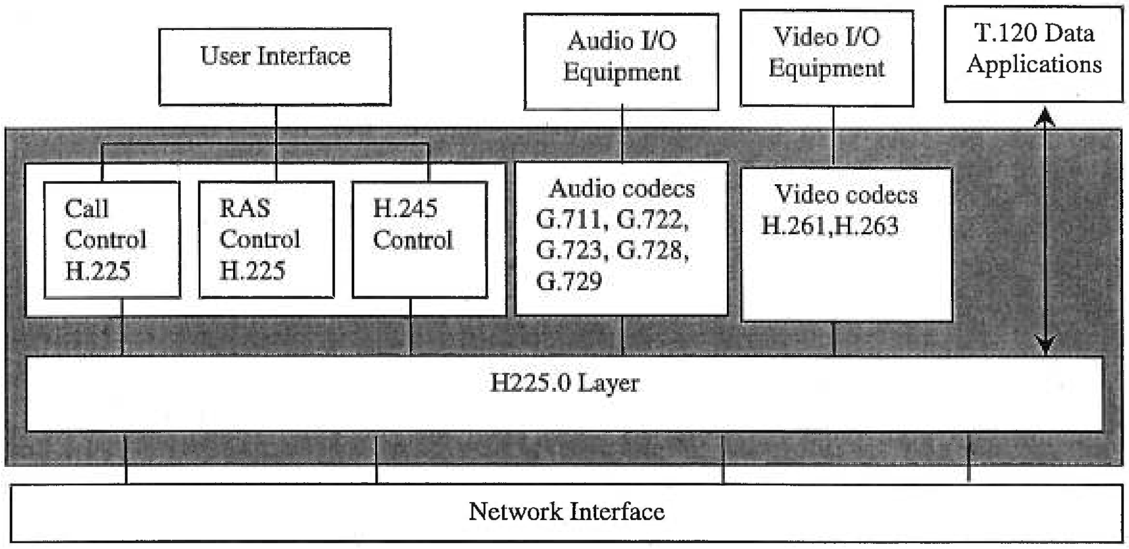


Figure 2.2: H.323 Terminal components

Video codecs

A video codec encodes video from a camera for transmission from an H.323 terminal, and decodes the received video code onto the video display on a receiving H.323 terminal. H.323 specifies the support of video as optional; hence the support for video codecs is optional as well. An H.323 terminal providing video must support the H.261 codec, which transmits near VHS-quality images at 64 kbps. The H.263 codec is optional, but is designed to be more resilient to packet loss and provides better image quality in low bandwidth environments. For this reason, it is often the default codec for applications running on the Internet.

H.225 Registration, Admission and Status

The H.323 terminal architecture defines a set of specific functions for Registration/Admission/Status (RAS), call signaling and control signaling. Registration, admission and status (RAS) is the protocol used for communication between endpoints (terminals and gateways) to gatekeepers. Functions of the RAS include performing registration, admission control, bandwidth changes, and procedures of disengagement between endpoints and gatekeepers. A RAS channel is opened between an endpoint and a gatekeeper for exchanging RAS messages.

H.245 Control Signaling

The H.245 standard provides the call control mechanism to enable H.323-compliant terminals to connect and communicate with each other [ITU-T H.245 96]. It specifies the control messages that carry information related to the opening and closing of logical channels to carry media streams, flow-control messages, general commands for requests and indications. The built-in framework of H.245 enables selections of codecs as well as capability negotiation such as bit rate, frame rate, and video format among H.323 endpoints.

H.225 Call Signaling

The H.225 call signaling is based on the Q.931 specification to establish a connection between two H.323 endpoints [Schulzrinne & Rosenberg 99]. This is achieved by exchanging H.225 protocol messages on a reliable call signaling channel. H.225 messages are exchanged directly between the endpoints if there is no gatekeeper in the H.323 network. This exchange of messages is called direct call signaling. If the messages are routed through a gatekeeper, it is referred to as gatekeeper-routed call signaling.

The H.225.0 Layer

The H.225 Layer formats the transmitted audio, video, data and control streams into messages for output to the network interface, using the Real Time Transport Protocol (RTP) and its associated control protocol, the Real Time Control Protocol (RTCP) [El-Marakby & Hutchison 96].

The T.120 standard

H.323 makes provision for supporting and using the T.120 standard so that data conferencing applications such as file transfer, application sharing, whiteboard and chat can operate in conjunction with H.323 connections [ITU-T T.120 96]. In this scenario, T.120 can use the H.225.0 layer to send and receive data packets, or simply create an association with the H.323 session, and use its own transport capabilities to transmit data directly to the network.

Developed by ITU-T, the T.120 standard comprises a suite of communication and application protocols that cover the document conferencing and application sharing portion of a multimedia conference. The T.120 recommendation specifies how to efficiently and reliably distribute files and graphical information in real-time and provides the framework for developers to create compatible products and services.

The T.120 standard is an « umbrella » standard that encompasses the communication and application standards described below:

T121: provides a generic application template specifying guidelines for building application protocols. To ensure application consistency, T.121 is a required standard for products developed under T.120 and the ITU recommends that non-standard applications incorporate T.121 to provide product interoperability.

T122: describes the multi-point services available to developers to enable users to send data as part of a conference. The description makes up the Multipoint Communication Services (MCS).

T123: specifies transport protocols such as the transportation and sequencing data, control of the flow of data and error-correcting mechanisms for accurate and reliable data delivery. The protocols described are for a range of networks, e.g. profiles are defined for TCP/IP and null modem connections.

T124: provides the generic conference control (GCC), which defines the application protocol supporting reservations and basic conference control services for multi-point teleconferences. Examples of such services are: directing people in and out of conferences, resource management when conference participants use multiple applications etc.

T.125: specifies the data transmission protocol by implementing the multipoint services defined by T.122.

T126: defines multi-point still image and annotation protocol: This includes collaborative data sharing, including whiteboard and image sharing, graphic display information, and image exchange in a multi-point conference.

T127: defines a method for applications to transmit files in a multi-point conference.

T128: specifies multi-point application-sharing protocols. It defines how users can share local applications such that other users can see the image of and collaborate on the shared application through the use of the mouse and keyboard to take control of the shared application as if it were running locally.

T134: provides a definition for a text chat protocol.

T135: defines conferencing reservation protocols in a T.120 environment, typically between client applications and a scheduling system which reserves resources for MCUs and bridges.

T140: implements the T.134 standard protocol for text chat within T.120.

ITU is also in the process of ratifying the T.136 and T.VMR standards which define remote device control/configuration, and Virtual Meeting Room controls respectively. Indeed, the ITU will continue working on new recommendations for T.120 as the market and technology evolves.

2.3.2 Gateways

A gateway is an optional entity in an H.323 conference. In general, the gateway reflects the characteristics of a packet-switched network endpoint to a switched-circuit network (SCN) endpoint, and vice versa. It provides a translation function between H.323 endpoints and other ITU-T compliant terminals such as transmission formats and communication procedures. It also performs call setup and clearing on both the network side and the SCN side. Terminals communicate with gateways using the H.245 and Q.931 protocols.

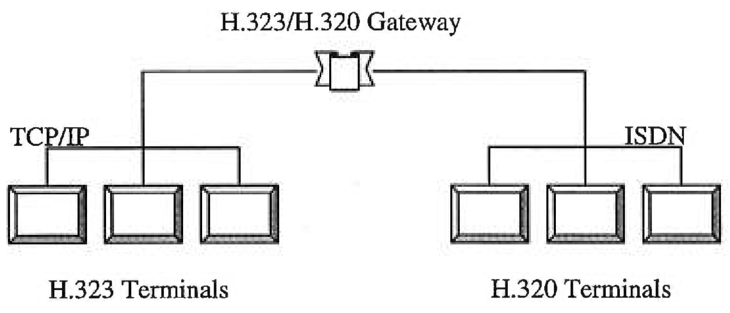


Figure 2.3: H.323 Gateway

Figure 2.3 illustrates an H.323 gateway that establishes a link between H.323 terminals and H.320 compliant terminals running over the ISDN switched-circuit network.

2.3.3 Gatekeepers

A gatekeeper is an optional element in an H.323 conference that provides call control services to H.323 endpoints. These services include address translation and resolution such as between an alias address (e.g. an email address) and a transport address, admission control using RAS messages, bandwidth control, and zone management [Toga & ElGebaly 98].

As illustrated in the following figure, the idea behind zone management is to define a scope of entities (terminals, gateways, and MCU's) under the control of a gatekeeper. It thus acts as the central point for all calls in its zone, and in many ways acts like a virtual switch.

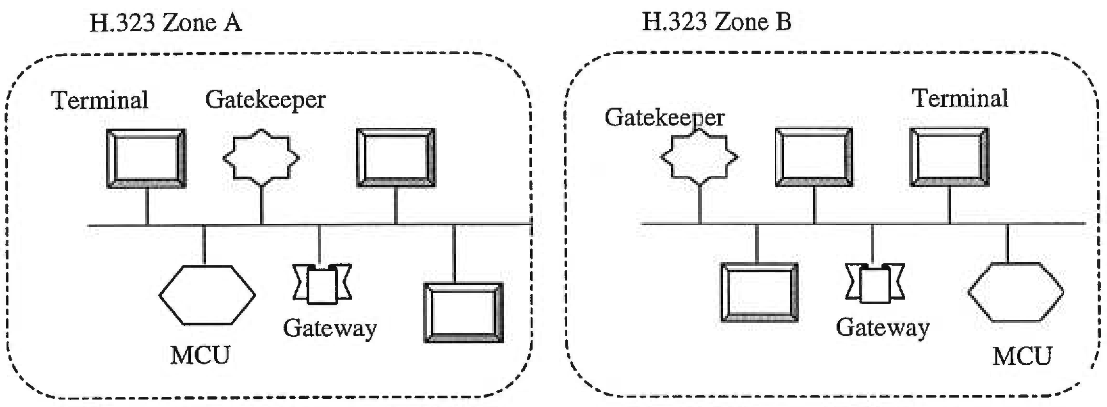


Figure 2.4: H.323 Gatekeepers and zones

The gatekeeper can also provide other optional functions such as call authorization, and accounting information. Some gatekeepers may also perform telephony service operations such as call forwarding and call transfer.

2.3.4 MCU

The MCU is an optional endpoint of a H.323 network which provides the support for three or more endpoints to participate in a multi-point conference. It may also connect two terminals in a point-to-point conference, which may afterwards develop in a multi-point conference. The MCU consists of a mandatory Multi-point Controller (MC) and optional Multi-point Processors (MP). The MC handles H.245 negotiations between terminals to achieve common capabilities for audio and video processing. It also controls conference resources by determining e.g., if audio and video streams are to be multicast. The MP allows mixing, switching or other processing of the media streams. Depending on the type of conference supported, it may process one or multiple media streams.

Centralized Versus Decentralized Multi-point Conferencing

The H.323 recommendation uses the concepts of centralized and decentralized multi-point conferences. These two models differ in their handling of the media streams and are illustrated in Figure 2.5.

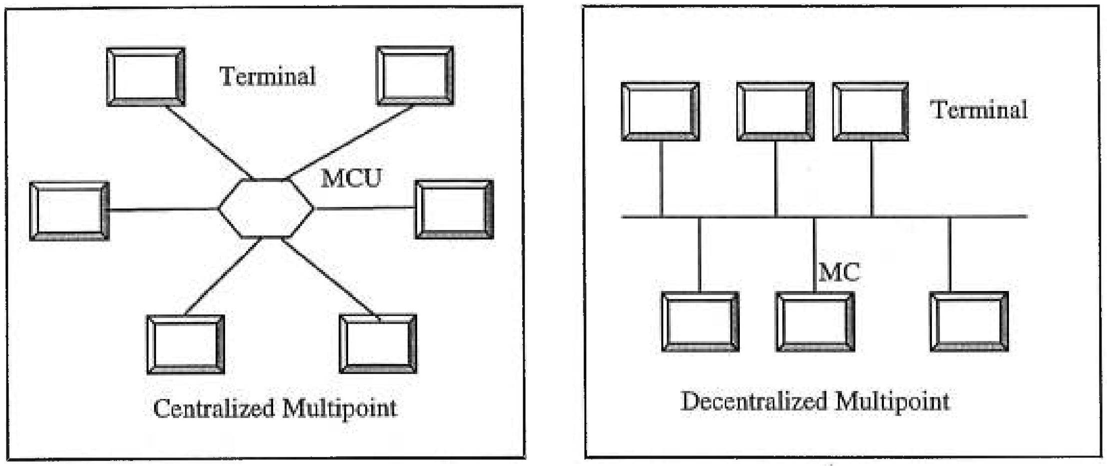


Figure 2.5: Centralized v/s Decentralized multipoint conferencing

Centralized multipoint conferences operate in the same fashion as circuit-based conferences, where the terminals send media streams in a point-to-point fashion to the MCU. The latter processes them and re-transmits the resulting streams to the terminals.

The logical topology of the decentralized model is a bus, and one of the endpoints on the network must contain an MC. The decentralized model shares similar control characteristics with its centralized counterpart but the media streams are handled differently. Each endpoint is responsible for its own audio mixing and video selection.

One advantage of the centralized model is that the endpoints need not be as powerful as in the decentralized model. Each endpoint only has to encode its locally generated media streams and decode the set received from the MCU. In parallel, this allows for endpoints connecting from different bit rates to participate in conferences. The MCU will in fact, choose the largest common set of attributes to operate the conference; e.g., it will operate in a least common denominator mode such as QCIF or a lower video resolution that CIF.

The decentralized model does not require the presence of an MCU which is a potentially expensive and limited resource. However, at all times, it requires that one of the participating endpoints contain an MC. Even if the endpoint containing the MC leaves the conference, the MC must stay active.

2.4 H.323 IP Conferencing

H.323 uses both reliable and unreliable communications. Control signals and data need reliable transport mechanisms because they must be received in the order they were sent, in a timely manner, and cannot be lost. On the other hand, delayed or lost audio and media streams may have little relevance to the user. This is why these media streams are usually carried using unreliable but efficient transport mechanisms.

In the TCP/IP stack, reliable transmission of messages are sent through the connection-oriented protocol Transport Control Protocol (TCP). TCP guarantees sequenced, error-free and flow-controlled transmission of packets. H.323 uses reliable end-to-end services for the H.245 Control Channel, the T.120 Channels for data transmission and the Call Signaling Channel.

The User Datagram Protocol (UDP) takes care of the unreliable services in the TCP/IP stack. It uses the connectionless or datagram mode of connection which only provides « best effort » delivery of packets. Delivery of packets is not guaranteed. H.323 uses UDP for the transmission of audio, video and the RAS channel.

2.5 RTP/RTCP

To cater for the QoS requirements of streaming of audio and video, the H.323 stack uses the Real Time Transport Protocol (RTP), designed by the Internet Engineering Task Force (IETF) [url IETF]. The moniker « transport protocol » in RTP could be misleading as RTP works together with TCP and mostly UDP, for the transport of audio and video streams. The name, however, implies that RTP is an end-to-end protocol and provides functions such as content identification of payload data, sequence numbering, time-stamping, media synchronization, and QoS monitoring of data transmission [Schulzrinne *et al.* 96]. RTP has no notion of a connection and can operate over connection-oriented or connectionless transport protocols. It thereby provides no reliability mechanisms. It also has no dependencies on particular address formats and requires that framing and segmentation be provided by lower layers.

RTP consists of two protocols: RTP for the real-time transmission of audio and video packets, and the Real-Time Control Protocol (RTCP) which caters for the control part.

2.5.1 RTP data packets

Continuous media like audio and video are encapsulated and carried into RTP data packets. Each data packet consists of a 12-byte header followed by the payload data which can either be a video frame or several audio samples. The header contains the following 5 principle fields:

Payload type:	a 1-byte field that identifies the kind of payload in the RTP data packet. For example, it can represent JPEG or H.261 video, or GSM audio.
Timestamp:	a 32-bit field that provides the generation instant the data packet is generated. It is mainly used for synchronization purposes and

	jitter calculation. The frequency of the timestamp depends on the payload type.
Sequence Number:	a 16-bit packet field that is incremented by one unit for each data packet sent. This can be used by receivers to detect packet loss and out-of-sequence packets within a series of packets with the same timestamp.
Marker bit:	the interpretation of a marker bit depends on the payload type. For video, the marker bit represents the end of a video frame; for audio, it represents the end of a talkspurt.
SSRC:	SSRC stands for "synchronization source identifier". It is a randomly generated 32-bit field that uniquely identifies the source of the RTP stream within a session. Typically, each stream in a RTP session has a distinct SSRC, and the probability that two streams have the same SSRC is very small.

2.5.2 RTCP feedback packets

RTCP is the control part of RTP which supports the protocol functionality. RTCP packets are transmitted by each participant to all other participants in an RTP session. The RTCP packets are distributed using IP Multicast. Figure 2.6 illustrates RTCP packets being sent periodically among participants using a multicast address for each RTP session opened.

RTCP packets do not encapsulate audio or video. Instead, they contain sender and/or receiver reports that contain information such as number of packets sent, number of packets lost and interarrival jitter. These are mainly used for monitoring the quality of service perceived by participants. The RTCP specification does not indicate what the application should do about the reports [El-Marakby & Hutchison 97]. It is up to the application developer to decide what is to be done with the feedback information.

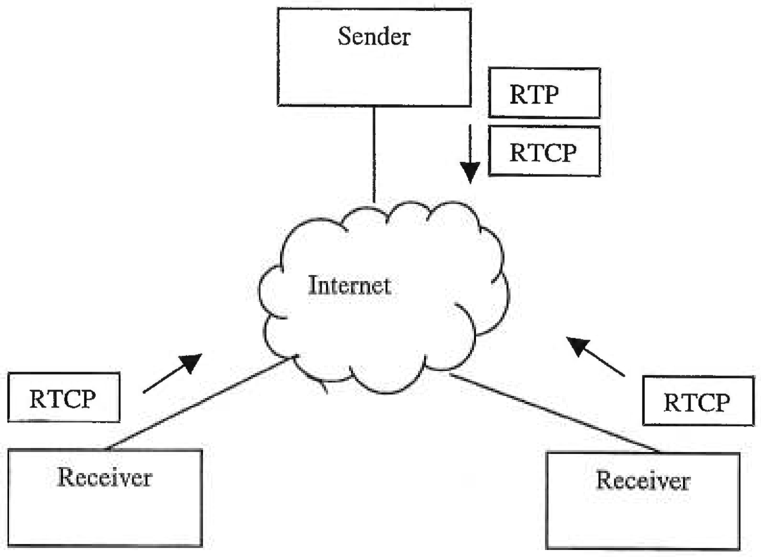


Figure 2.6: RTP/RTCP packet flow

If RTP packets are encapsulated in UDP datagrams, the data and control packets use two consecutive ports; the data port being the lower, even numbered one. If RTP works on top of other protocols such as ATM AAL5 [url ATM], it is possible to carry both data and control packets in a single protocol data unit, with the control followed by the data.

Receiver reports and packets

A reception report is generated at each receiver for each received RTP stream in a session. The reception reports are then aggregated or "stacked" into a single reception packet which is periodically sent to all other participants through multicast. A reception report contains the following principle fields:

- The SSRC of the RTP stream for which the reception report is being generated.
- The fraction of packets lost within the RTP stream. The receiver divides the number of packets lost by the number of packets sent as part of the stream.
- The last sequence number received from the last RTP stream.
- The interarrival jitter, which is calculated as the average interarrival time between successive packets in the RTP stream.

Since these reception packets are multicast, all session members can survey how the other participants are faring. The sender can adjust its transmission rates after analysis of the reception packets received based on its congestion control policy.

Sender report packets

The sender generates and transmits to other participants a sender report packet for each RTP stream in a session. This report packet contains the following main information:

- The SSRC of the RTP stream.
- The wall-clock time and timestamp of the most recently generated packet in the stream. These two values are used for the play-out synchronization of different media e.g., lip-synchronizing of audio and video.
- The number of packets sent in the stream.
- The number of bytes sent in the stream.

Source description packets

The sender also generates source description packets used for identifying itself. These packets contain textual information such as the sender's canonical name, his/her email address, telephone number and other relevant information. The SSRC of the associated RTP stream is also included in order to provide a mapping between the host name and the source identifier SSRC.

2.5.3 QoS parameter calculation

The calculation of QoS indicators such as throughput, delay and jitter can be performed by the information contained in the sender reports. Timestamps in RTP/RTCP are based on the Network Time Protocol (NTP) standard [Mills 92]. NTP is an Internet protocol used to synchronise computer clocks to some time reference. RTCP uses two principle timestamps in the Sender report namely the Last Sender

Report (LSR) and the Delay Since Last Sender Report (DLSR) timestamps for throughput calculation and the "Interarrival jitter" field for jitter calculation.

LSR and DLSR

LSR is a 32-bit timestamp received as part of the most recent RTCP sender report from the source. DLSR is the delay, expressed in units of 1/65536 seconds, between receiving the last sender report from the source and the sending of the receiver report. Figure 2.7 shows the relation between LSR and DLSR and how they are used.

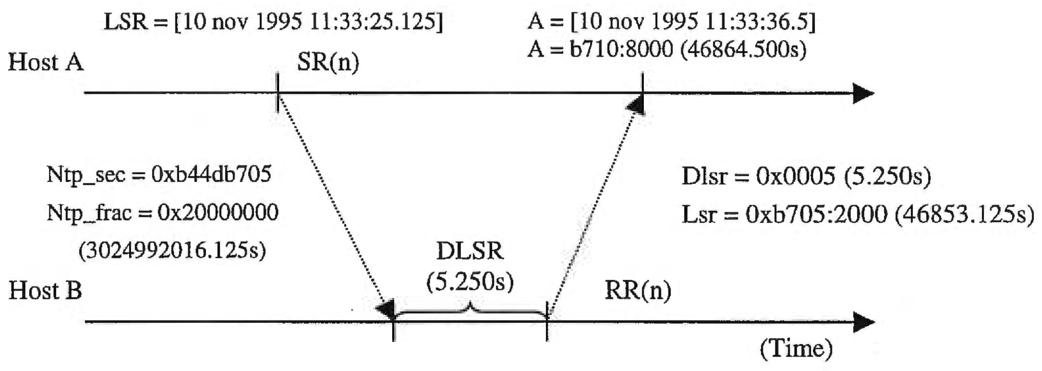


Figure 2.7: LSR and DLSR calculation [Schulzrinne *et al.* 96]

Two hosts A and B communicating, with A being the sender and B the client. From the figure, the round trip time (RTT) can be computed as follows:

$$RTT = A - (DLSR + LSR)$$

The approximate time for the RTCP packet to travel from host B to host A is approximately RTT/2. Hence, the throughput (T_{RTCP}) of the link for an RTCP packet can be calculated as follows:

$$T_{RTCP} = RTCP_{AV} / (RTT/2)$$

where $RTCP_{AV}$ is the average size of the receiver report. Therefore, the throughput of RTP packets can be estimated as

$$T_{RTP} = (RTP_{AV} * T_{RTCP}) / RTP_{AV}$$

where RTP_{AV} is the average size of an RTP packet.

Inter-arrival jitter

The inter-arrival jitter gives an estimate of the statistical variance of RTP packet inter-arrival time. The inter-arrival value is the mean deviation of the difference (D) in packet spacing time. For two consecutive arriving packets i and j, this difference can be calculated as:

$$D_{(i,j)} = (R_j - R_i) - (S_j - S_i)$$

where R_i is the time of arrival of packet i in timestamp units, and S_i is the RTP timestamp from packet i. The above calculation is performed for each RTP packet received.

The jitter (J) is then calculated as thus:

$$J = J + (| D(i-1,i) | - J) / 16$$

The jitter value is sampled for every reception report that is issued. The 1/16 factor is used for noise reduction while keeping a reasonable convergence rate.

2.5.4 RTCP scaling problem

Since all session members periodically send RTCP packets, this can lead to a potential scaling problem. The aggregate transmission rate of RTCP packets can by far exceed the rate of RTP packets sent by the sender. There is a need to balance the desire for up-to-date control information against the desire to limit the control traffic as a small fraction of data traffic. RTCP thus attempts to scale the control traffic load by limiting it to 5% of the session bandwidth [Rosenberg & Schulzrinne 98].

In this chapter, we covered one important aspect of the proposed architecture that addresses its interoperability and compatibility with other industry applications. We presented the H.323 recommendation which is one popular standard for carrying multimedia data over packet switched networks, and which lays the foundation for communication among multimedia applications. Our proposed architecture has been designed to be H.323-compliant. In the next chapter, we address another crucial part of our work with respect to Quality of Service. We give the state of the art as regards QoS at the core level of the Internet and existing QoS architectures. We have applied many of the techniques used in these areas on our architecture.

Chapter 3

The Quest for QoS

3.1 Defining QoS

The term quality of service (QoS) is often used to represent "the set of qualitative and quantitative characteristics of a multimedia application that fulfills the functionality required from the application" [Vogel *et al.* 95]. It consists of performance-related attributes such as video transmission rate and delay; format-related attributes such as video frame rate and resolution; cost-oriented attributes such as billing and copyright charges; and finally, user-oriented attributes such as subjective image and sound quality.

Much work has been conducted in the context of standardized frameworks and models to provide QoS within distributed systems. QoS management in this area has mainly focused in finding appropriate QoS characteristics for the different system components in a distributed multimedia application and reserving the corresponding resources to meet user desired QoS and optimize system performance. This has resulted in several QoS architectures being defined in the literature.

A considerable amount of research has also been done on QoS management for communication networks. This work has concentrated mainly on resource allocation in the networks to assure specific QoS characteristics for new connections on ATM networks [Bochmann & Hafid 96]. Recently, there has been a growing interest in the provision of QoS at the core of the Internet. With the rapid transformation of the

Internet into a commercial infrastructure, major router/switch vendors are increasingly providing QoS mechanisms in their high-end products.

3.2 Internet QoS

Since the late 1980's, the Internet community has been working on ways to accommodate real-time applications with strict delay requirements such as conferencing systems. Today's Internet can only provide a "best effort" service with no guarantees of timeliness and delivery. Hence, QoS cannot be guaranteed. In parallel, while today all traffic through the Internet gets treated equally, it is becoming apparent that there is a need to give preferential treatment to some application traffic. Users requiring videoconferencing with guaranteed QoS will likely be charged more than those requiring best-effort QoS. There are presently two schools of thought debating the question whether there is really a need for mechanisms to provide QoS. Some opinions believe that high enough bandwidth can be automatically and cheaply achieved through the use of optical fibers and wavelength-division multiplexing (WDM), hence guaranteeing QoS. On the other hand, there is the belief that no matter how much bandwidth networks can provide, new demanding applications will always consume the available bandwidth; hence the need to provide mechanisms to meet QoS [Xiao & Ni 99]. In the light of our work, we strongly favor the last opinion and believe that mechanisms should be implemented to provide reasonable service to applications.

The Internet Engineering Task Force (IETF) has worked on some service models and mechanisms for meeting the demand for QoS. Among the most notable are the Integrated Services model (IntServ), the Differentiated Services model (DiffServ) and Multi-protocol Label Switching (MPLS).

3.2.1 The Integrated Services model (IntServ)

Proposed in the early 1990's under RFC 1633, the Integrated Services model [Braden *et al.* 94], mostly known as IntServ, encompasses two service classes together with the best effort service. These service classes are:

- The Guaranteed Service for applications with fixed delay bound requirements [Shenker *et al.* 97].
- The Controlled-load Service for applications requiring reliable and enhanced best effort service [Wroclawski 97].

The philosophy behind the IntServ model is that routers must be able to reserve resources in order to provide special QoS for specific packet streams or flows. To reserve resources, the Resource Reservation Protocol (RSVP) was defined [Braden *et al.* 97]. RSVP is a network-control signaling protocol that allows applications on the Internet request specific QoS treatment of their data flows. The signaling process of RSVP is illustrated in the Figure 3.1.

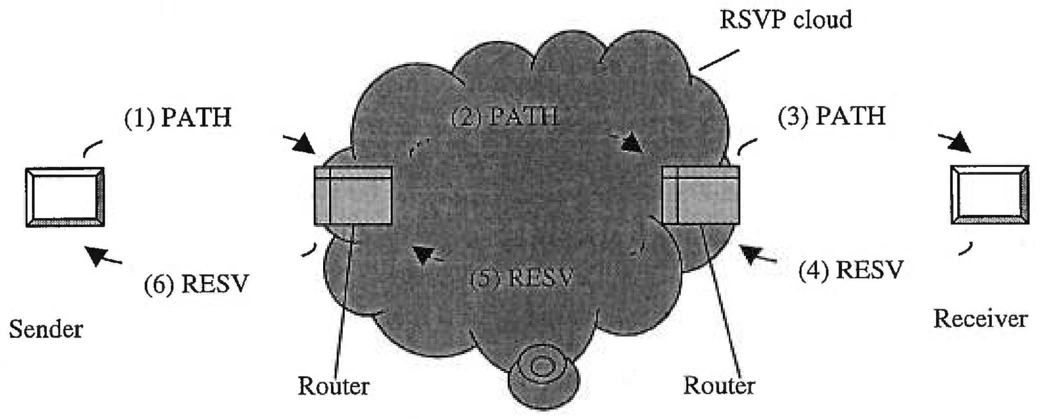


Figure 3.1: RSVP signaling

A PATH message containing the characteristics of the traffic is sent from the sender to the receiver. Each router along the path forwards the PATH message to the next hop as determined by the routing protocol. Once the receiver host receives the PATH message, it responds with a RESV message that is used to request resources for the flow. Every router along the path can accept or reject the request of the RESV message depending on whether it has enough resources to accommodate the flow without affecting earlier reservations. If the request is accepted, link bandwidth and buffer space are allocated in the router for the flow; if the request is rejected, an error message is sent to the receiver and any reservation made along the path for that flow are removed. The latest version of RSVP can reserve resources for an aggregation of flows and setup explicit routes (ERs) with QoS requirements.

Besides having the signaling protocol and the admission control for granting or rejecting requests, routers in the IntServ model use two other components namely; the classifier routine, which classifies packets on arrival at the router, depending on the service requested; and the scheduling routine which schedules packets based on their QoS requirements such as playout deadline.

The IntServ model faces an obvious scaling problem. The amount of state information that needs to be maintained in routers can become logistically huge as the number of flows increases. This places a high requirement on routers. In addition, for the model to be properly deployed, all routers will need to provide for RSVP, admission control, packet classification and scheduling, which is very far from the actual state of Internet routers presently.

Another approach entitled the Differentiated Services was then pursued by the Internet community.

3.2.2 The Differentiated Services model (DiffServ)

The Differentiated Services model, most commonly known as the DiffServ model [Blake *et al.* 98], is largely based on the type-of-service (TOS) field within the IPv4 header. Designers of the IP header came up with this one-byte field as a way classify datagrams. Three bits of the TOS field, known as the IP precedence bits can be set to indicate low-delay, high throughput or low loss-rate service [Nichols *et al.* 98]. Because of the limited choice, the use of these bits was very rare in early routers.

Some recent commercial routers found some good uses to these bits; e.g. to differentiate between high and low priority traffic, although without using their significance as initially meant. The DiffServ model renames the TOS field as the DS field and redefines its layout. It also provides a set of packet forwarding treatments known as per-hop behaviors or PHBs. The basic working of DiffServ can best be explained by using a scenario.

Figure 3.2 illustrates a WAN involving 4 hosts (H1, H2, H3, and H4), 4 leaf routers (R1, R2, R6, and R7) and 3 core routers (R3, R4, and R5). Host H1 is sending packets to host H3 and host H2 is sending packets to host H4.

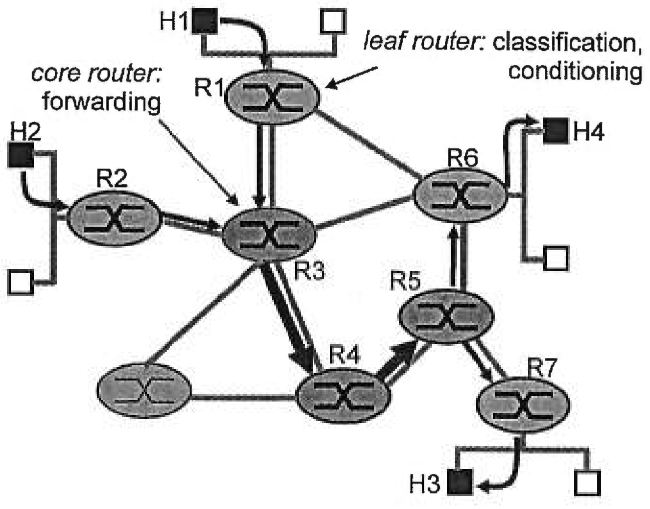


Figure 3.2: The DiffServ model [Kurose & Ross 99]

The DiffServ model proposes two sets of functional elements.

- Ingress packet classification: At the ingress of the network involving any of the Differentiated Services-capable hosts or routers, the DS fields of arriving packets are marked. For example, packets flowing from H1 to H3 will be marked at router R1 while those flowing from H2 to H4 will be marked at router R2. The mark received by a packet identifies the class of service or "behavior aggregate" to which it belongs. Once marked, the packets can be immediately forwarded, or delayed before being forwarded or simply discarded based on some policy agreed between the hosts and the service provider.

- Core network packet forwarding or PHB's: The markings done at the ingress of the network will determine how they will be forwarded into the core network. In our example, if packets from H1 to H3 receive the same markings as those from H2 to H4, core router R3 will treat these packets as aggregates and forward them without distinction. RFC 2475 defines a PHB as a "description of the externally observable forwarding behavior of a DS node applied to a particular DS behavior aggregate" [Blake *et al.* 98]. More simply, the role of the PHB is to prioritize among the classes of traffic, e.g., whether one class of traffic receives priority over another one. It does so by influencing the sharing of a router's buffers and link bandwidth among the competing classes of traffic.

Expedited Forwarding (EF) PHB v/s Assured Forwarding (AF) PHB

So far, the IETF has been working on two PHB's namely the Expedited Forwarding PHB, and the Assured Forwarding PHB. The first PHB guarantees an aggregate class of traffic enough bandwidth so that the output rate of the traffic equals or exceeds the minimum configured rate. The Assured Forwarding PHB is more complex and divides traffic into four classes where each AF class is guaranteed some minimum amount of bandwidth and buffering. For each class, packets are partitioned into "drop preference" categories so that if ever congestion occurs within one AF class, a router can discard packets based on their drop preference values. By varying the amount of resources allocated to each class, different levels of performance can be provided to different AF traffic classes.

Service Level Agreements (SLA)

In order for a customer to have differentiated services, it must have a service level agreement (SLA) with its ISP. The SLA will define the service classes e.g., premium service (for low-delay and low-jitter applications) or assured service (applications requiring better than best effort) or olympic service. While the DiffServ model only defines the DS fields and PHB's, it is up to the ISP to decide which services to provide [Xiao & Ni 99].

The DiffServ model has the advantage of being more scalable than the IntServ model since the amount of state information varies with the number of classes of service rather than the number of flows. Also, classification and shaping operations are only needed at the ingress routers.

3.2.3 Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) is the latest step in the evolution of routing/forwarding and multi-layer switching technology for the core of the Internet [Rosen *et al.* 98]. It emerged from the IETF's effort to standardize a set of proprietary multilayer switching solutions developed in the mid 1990's. MPLS is called *multiprotocol* because it works with the Internet Protocol, Asynchronous Transport Mode (ATM), and frame relay network protocols. In the OSI model, MPLS is between the Link Layer (L2) and the Network Layer (L3).

The motivation for MPLS is to use a fixed-length label to decided packet handling. Each MPLS packet has a header consisting of the following fields:

- The 20-bit Label field that carries the actual value of the MPLS label.
- The 3-bit Class of Service (COS) field, which can affect the queuing and discard algorithms applied to the packet as it is transmitted through the network.
- The 1-bit Stack field to support a hierarchical label stack.
- The 8-bit Time to Live (TTL) field that provides conventional IP TTL functionality.

This MPLS header is encapsulated between the Link Layer header and the Network Layer header. MPLS uses a control-driven model (triggered by control traffic such as routing updates) to initiate the assignment and distribution of label bindings for establishing label-switched paths (LSP's). An LSP is created by concatenating one or more label switched hops, allowing a packet to be forwarded from one label-switching router (LSR) to another across the MPLS domain. The LSR's will negotiate the semantics of each label, i.e., how to handle a packet with a particular label from the peer.

The MPLS control component is based on IP functionality which itself, is based on proprietary multi-layer switching solutions. MPLS however, defines new standards-based IP signaling and label distribution protocols in order to support multi-vendor interoperability. Figure 3.3 illustrates the working behind MPLS.

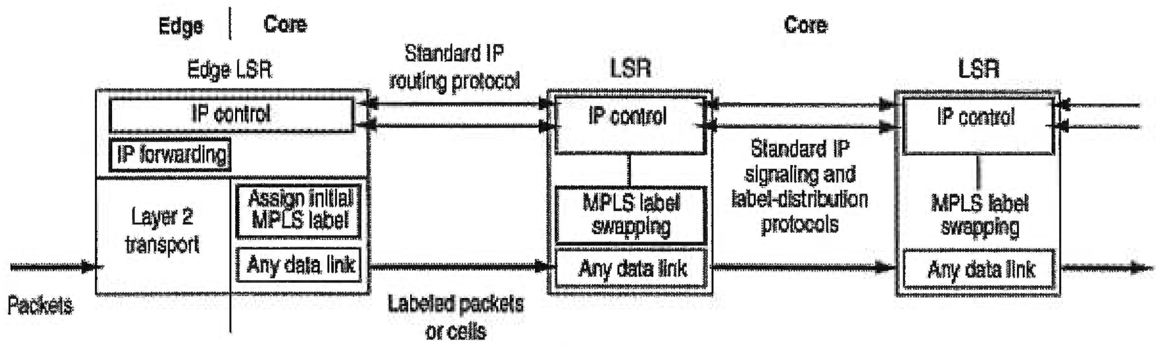


Figure 3.3: The MPLS model [Semeria 00]

MPLS headers are inserted at the ingress routers. Core LSR's use the MPLS label of the packet as an index to look in a forwarding table. This process is comparatively faster than the parsing process of the routing table as done in conventional IP routing [Nilsson & Karlsson 97]. The forwarding table contains entries that specify how to process packets carrying the indexing label. It is constructed from the label distribution and indexed by labels. The packet is processed as specified by the forwarding table entry. The incoming label is replaced by the outgoing label before the packet is switched to the next LSR. This label-swapping process is similar to VPI/VCI processing in ATM networks. Before a packet leaves the MPLS domain, its MPLS label is removed.

The LSP's can be used as tunnels. Once the LSP's are setup, the path of a packet path can be completely determined by the label assigned by the ingress LSR. Compared to other tunneling mechanisms, MPLS is unique in that it can control the complete path of the packet without specifying the intermediate routers.

3.2.4 Comparison of the three models

An interesting question is what is the best mechanism for delivering QoS on the Internet? Although significantly different in nature, the three models presented can work complementarily with each other. For example, some router implementations use RSVP to simply reserve bandwidth without managing traffic and leave the traffic management to DiffServ. ISP's could use RSVP to reserve 10 Mbps of bandwidth and then tunnel their DiffServ-tagged traffic over that pipe.

Proponents of MPLS require that vendor implementations of MPLS be compatible with the IntServ model, including RSVP. After all, RSVP could be extended in order to perform the label distribution of MPLS instead of the more common label distribution protocol (LDP). Finally, MPLS and DiffServ can work together to provide QoS. There are some minor changes in the way the packets are processed, but the architecture proposed [Xiao & Ni 99] has been shown to be very scalable.

Many opinions question the use of RSVP since resource reservation represents a fundamental change in the architecture of the Internet. It also faces a scaling problem that makes it quite unsuitable for the Internet. This is one reason RSVP has been shunned by ISP's and seems more confined to enterprise networks. The problem with MPLS is that it is at least a year away from implementation in ISP networks. The technology, however, has the attention of many ISP's. With the growing level of interest many industry professionals are gearing towards MPLS, we believe that at least in the medium term, among the techniques presented MPLS will be the preferred deployed mechanism to provide QoS at the core level of the Internet.

3.3 QoS Architectures

Initially, QoS was considered at individual system components, such as the operating system and network. This resulted into frameworks that had the following limitations: they offered few QoS oriented configurable interfaces and it was difficult to reconcile the existing QoS notion at different system levels and among different network architectures.

As a result of these limitations, a number of QoS architectures has been proposed by various research groups. Their objectives are mainly to define a set of QoS configurable interfaces that formalize QoS in end-systems and networks by providing a framework that integrates QoS control and management mechanisms [Hafid & Bochmann 97a]. We describe some of these architectures in the following sections.

3.3.1 The OMEGA Architecture

OMEGA is a QoS architecture for provision of real-time guarantees at endpoints in a network multimedia system [Nahrstedt & Smith 95a]. It has been developed at the University of Pennsylvania. It is based on the integration of a resource management model and a communication model. The following figure illustrates the

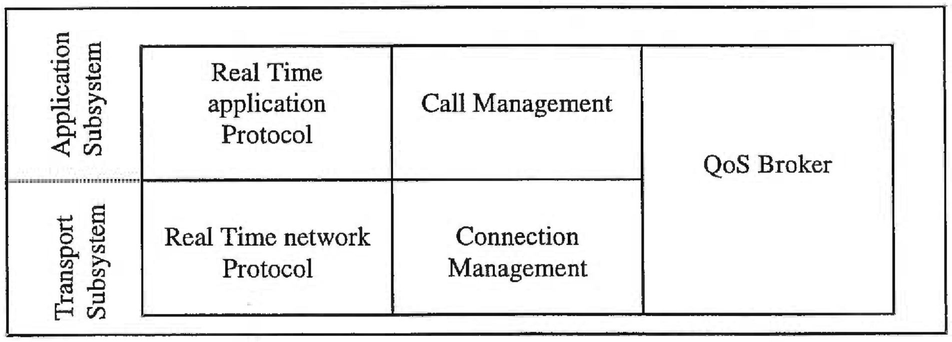


Figure 3.4: OMEGA communication model

communication model. It is a two-layer system composed of the transport subsystem and the application subsystem.

The transport subsystem provides transport and network services such as connection establishment and termination, and data movement from/to application ring buffers to/from the network host interface. This is implemented by the Real-Time Network Protocol (RTNP).

The application subsystem provides services to support application requirements. Such services include call management, synchronization, media delivery and device management such as input/output device rate control. They are implemented using the Real-Time Application Protocol (RTAP).

Both sub-systems provide communication service guarantees over specified communication channels. Such guarantees are negotiated by the QoS broker, which is the main component of the architecture [Nahrstedt & Smith 95b]. It translates the requirements of the application and negotiates the allocation of resources with the operating system and the network.

Resource reservation is based on QoS parameters associated with the specific media device. A translator is used to obtain lower-level QoS requirements from the user-defined application-level parameters, adopting a fixed set of translation relations for each media type. This two-way translation allows dynamic changes in resource reservations to be reported to the user as application-level QoS parameters. The obtained parameters result in the reservation of network and operating system resources such as CPU slots and buffers, both local and remote.

The local QoS Broker aiming to perform reservations of remote resources is responsible for interacting with other QoS Brokers located remotely. Proponents of this architecture have devised a buyer/seller protocol to achieve the above purpose. The QoS Broker also employs orchestration services and relies on information stored in resource databases balance the resource allocation among multimedia devices, operating systems and the network.

3.3.2 The Extended Reference Model (XRM)

The Extended Reference Model (XRM) [Lazar *et al.* 96] was designed at the University of Columbia by the COMET Group. It models the communications architecture of networking and multimedia computing platforms consisting of three components namely: the Broadband network, the Multimedia network and the Services and Applications network. An overview of XRM is depicted in Figure 3.5.

The Broadband network is defined as the physical network which consists of switching and communication equipment and multimedia end-devices. Upon this physical infrastructure resides the Multimedia network whose primary task is to provide support to realize services with end-to-end QoS guarantees. This is achieved by building upon a set of QoS abstractions derived from the Broadband network.

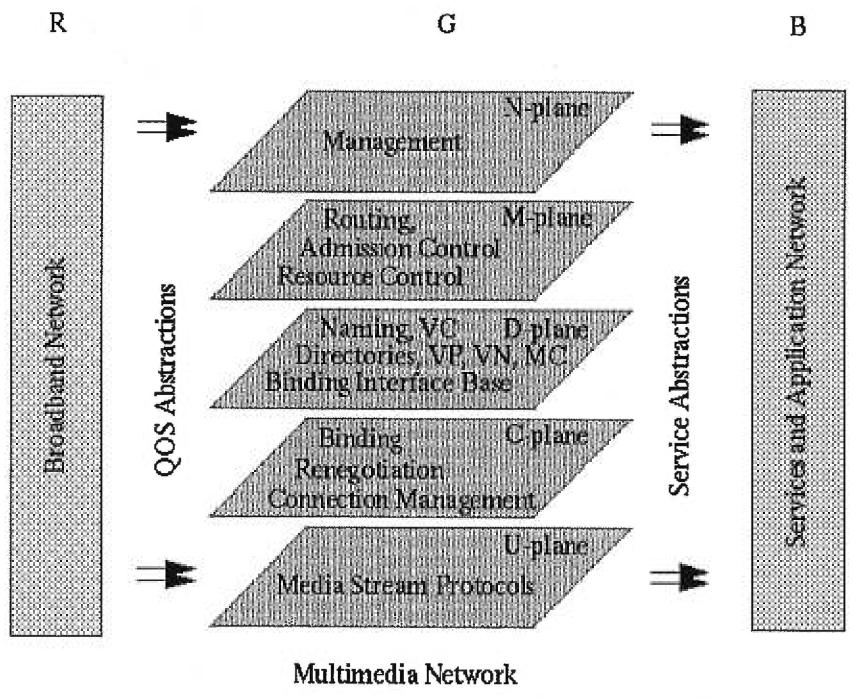


Figure 3.5: XRM architecture [Lazar *et al.* 96]

This set of QoS abstractions define the resource management and control space, and the process of service creation calls for resource reservation. The Multimedia Network thus provides a programming model that allows service behavior to be specified and executed. The user can only access the service abstractions provided by the Services and Application Network level.

The services visible at the Multimedia Network level are provided by xbind, a flexible programming platform for creating, deploying and managing multimedia devices. Xbind can be viewed as a broadband kernel for multimedia networks that provides end-to-end QoS. It includes software components for implementing mechanisms for distributed network resource allocation, broadband signalling, real-time switch control, multimedia transport and device management. It is based on the Common Object Request Broker Architecture (CORBA), an industry standard distributed object-oriented platform [OMG CORBA 96]. This architecture presents the service provider with a set of open binding interfaces with multimedia and QoS capabilities.

3.3.3 QoS-A

The Quality of Service Architecure (QoS-A) [Campbell *et al.* 94] was developed by the Distributed Multimedia Research Group (DMRG) at the University of Lancaster. It offers a framework for specifying and implementing the required performance properties of multimedia applications over ATM networks.

The QoS mechanisms offers system-wide guarantees i.e., they span the application platform and operating system, end-systems, the transport sub-system and network. Proponents of the architecture incorporate the following notions:

- A flow, which characterizes the handling such as the production, transmission and consumption of a media stream as an integrated activity.
- A service contract, which is an agreement between the user and the service provider regarding the QoS guarantees to be provided.

- Flow management, which represents the monitoring and maintenance of the contracted QoS levels.

QoS-A allows the provision of data flows with an associated QoS through the integration of devices, end-systems and the network. QoS-A constitutes of a structure of layers and planes integrated with ATM to provide a QoS-configurable communication mechanism. The three planes that compose the architecture are shown in Figure 3.6. The Protocol Plane is responsible for data transfer. It is subdivided in a user plane and the control plane. The user plane performs the transmission of media data while the control plane takes care of control data transmission. The QoS Maintenance Plane performs the monitoring and maintenance of the QoS levels specified in the service contracts. Based on the monitored statistics from the system, this plane configures the available resources to provide the specified QoS levels. The Flow Management Plane is responsible for flow establishment procedures such as admission control and resource reservation, QoS mapping and adaptation.

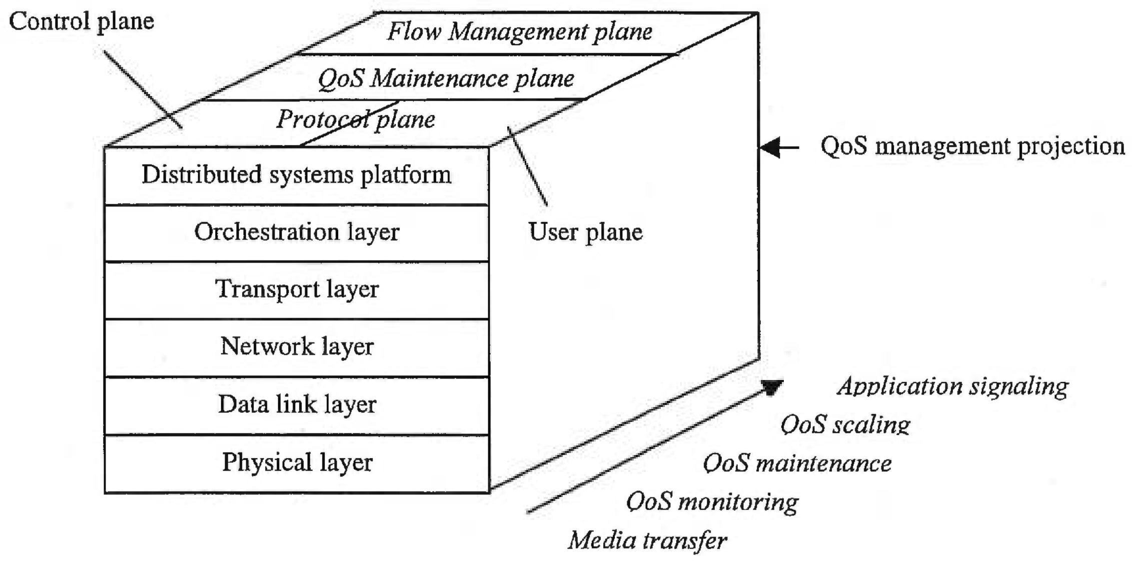


Figure 3.6: QoS-A architecture

The Protocol and QoS Maintenance planes are subdivided into four layers namely:

- The distributed systems platform, which is compatible with ODP and provides services for multimedia communication, QoS configuration and specification.
- The orchestration layer, which is responsible for jitter control and rate regulation of media streams. It also provides mechanisms for media synchronization.
- The transport layer, which provides a basic QoS-configurable communication service.
- The network and lower layers, which provide the basis for the communication service.

QoS-A is implemented over an enhanced Chorus micro-kernel and enhanced protocols for multimedia communications built on a local ATM network.

3.3.4 Comparison of the QoS architectures

All the architectures described focus on communication systems and operating systems to guarantee QoS. They concentrate on resource reservation and real-time scheduling mechanisms. All the entities involved in QoS support are known before the connection set-up phase.

QoS-A and XRM provide service guarantees with respect to network and end-system resources by integrating the support of resource management in the network and end-systems. The matter of translation and mapping is simplified because there is no need for QoS information to be specified at a higher level of abstraction. Real-time scheduling policies and buffer reservation schemes are used to perform the functions for supporting the processing of streams. Such functions include fragmentation and error control, stream synchronization and data movement.

The OMEGA architecture, on the other hand, provides QoS guarantees on an application-to-application basis. The resource management at both end-systems and the network are based on user requirements specified using high-level QoS parameters, and not QoS requirements of a single connection. Examples of these high-level parameters are frame rate, video color etc. The OMEGA architecture provides mechanisms to negotiate during the call establishment phase, the QOS guarantees of all incoming and outcoming connections of the application.

[Vogel *et al.* 95] provide a survey and comparison of other architectures used in the literature.

Much of our work was inspired of the strengths of the architectures presented. Our architecture heavily relies on adaptation techniques and scheduling techniques for QoS provision, as does the QoS-A architecture. In the next chapter, we describe current implementations of adaptation mechanisms, and how they can be used in our architecture for QoS provision.

Chapter 4

Adaptive Systems

4.1 Principles of QoS Management

To ensure that the QoS requirements of users are met, there is a need for a certain level of QoS management to exist in the system. QoS management functions can be defined as a set of activities that work together in order to satisfy the requirements of the user [Hutchison *et al.* 94]. QoS adaptation is one of the several activities of QoS management. Its role is to maintain, as far as possible, the continuity of a service when the initial contracted QoS is no longer possible [Huard *et al.* 96].

Before understanding how QoS adaptation can be implemented, it is important to understand the other activities that QoS management comprises of. These activities are namely: QoS specification, QoS mapping, QoS negotiation, resource reservation, QoS monitoring, QoS policing, QoS accounting, admission control and QoS termination. These functions are briefly described below.

4.1.1 QoS specification

The desired quality of service of users has to be conveyed in terms that are familiar to them. Examples of such requirements can be the video frame rate, video resolution, and even subjective attributes such as the sound quality. The users specify values for these QoS parameters; usually, these take the form of lower and upper limits representing the absolute minimum and the desired QoS. These parameters are to be monitored at all times, and if the minimum threshold value cannot be met, the service is often aborted (whether automatically or by notifying the user). In some cases, a maximal value is introduced. This is relevant for cost reasons; the user would not want to violate this value orelse he/she will be charged a higher cost.

Other research works have proposed a class classification for QoS specification [Schulzrinne *et al.* 90]. Those classes divide application requirements to reliable, unreliable, time dependent and time independent. This approach seems quite inflexible, coarse and limited in the use of class choices.

Another approach consists of using atomicity relationships for the user to specify preferences or priorities on media streams [Ravindram & Steinmetz 93]. Atomic logical operators such as AND, OR and NOT can be used in relationships such as "(audio stream AND video stream) OR audio stream" to specify that the delivery of audio and video streams is desired although the delivery of only audio stream is accepted.

4.1.2 QoS mapping

The QoS mapping activity provides a translation between QoS representations at different adjacent system levels. It is important that the service provider properly interprets QoS parameters as defined at the user level. For example, while the user will probably be asking for a frame rate specification, the latter will have to be mapped on resource parameters such as bandwidth, buffers or CPU time; measures which are more apt to be handled and managed at the network and end-system levels. Other studies have focused on direct mapping of QoS parameters from one layer to the next; e.g., the mapping of transport layer parameters into network parameters.

4.1.3 QoS negotiation

The QoS negotiation activity attempts to find an agreement on the required values for the QoS parameters for a session between the user and the system. It tries to satisfy the user requirements by taking into consideration the latter's QoS specifications. If it can meet the requirements at a cost that is acceptable by both parties, a connection setup between the two can be started. Otherwise, the currently proposed parameters are rejected and there is usually room for a re-negotiation [Hafid *et al.* 97].

4.1.4 Resource Reservation

Applications attempt to provide a guaranteed level of QoS by reserving a certain amount of end-system resources and/or network resources. These resources can take the form of CPU time and slots, buffer space, memory, bus and network bandwidth. Reservations are made via signaling calls that carry the user QoS requirements. The mechanisms proposed in the literature depend on the traffic models and scheduling algorithms used in the system as well as the characteristics of the system e.g., if it is single processor or LAN [Nahrstedt 95]. One example of a traffic model is the linear bounded process arrival time (LBAP) [Cruz 91]; examples of real-time scheduling algorithms are the stop-and-go queuing, the hierarchical round robin and the weighted-fair queuing algorithms [Schulzrinne *et al.* 94].

4.1.5 QoS Monitoring

The QoS monitoring activity consists of providing appropriate methods and procedures of measuring the actual performance of attributes during a session, so as to detect any violation on the agreed QoS and hence notify the system and the user. The violations could result from traffic delay on network, packet loss etc. Critical issues in QoS monitoring are which measured parameters to take into consideration (subjective attributes such as sound quality are discarded in the measurement process), the frequency of the measurements and their time interval. Protocols such as RTP/RTCP provide monitoring facilities in terms of delay, jitter, loss rate and bandwidth measurements.

4.1.6 QoS policing

QoS policing represents the set of actions that are taken so that the the integrity of the system is maintained. Such actions might become necessary following QoS violations or a breach in contract between the user and the system. For example, if the agreed throughput is 1 Mb/sec and the user is sending at 2 Mb/sec, the system might enforce the policy that the additional 1 Mb/sec is discarded. That will also ensure that the QoS of other concurrent connections are maintained too. To meet up with user violations, QoS policing actions usually include packet dropping and/or cost increases.

4.1.7 QoS Accounting

QoS Accounting takes into account the cost charge to the user for the amount and quality of the service used. Without cost constraints, it is obvious that customers will ask for the best possible QoS resulting in bottlenecks of the network. Several cost components have been devised including: cost per service, cost of duration, time period of service use, cost by security levels. The cost factor is becoming more and more pertinent particularly in view of the commercial infrastructure the Internet is becoming. The problem of cost calculation elevates dramatically whenever the service provided spans multiple networks, in which case the tariffication might be quite complex.

4.1.8 Admission Control

The role of admission control is to accept or reject a user request for a service. It does so by comparing the resource requirements associated with the request and the available resources in the system. The policies guiding admission control usually depend on resource availability, the system scheduling mechanisms and the data traffic characteristics resulting from the new request. The admission control algorithm must ensure that acceptance of the request will in no circumstance jeopardize any guaranteed QoS to previous connections. [Ferrai *et al.* 92] define a set of admission control tests using a traffic model based on the minimum packet-inter-arrival time and the maximum packet size and based on a singular node running the earliest deadline first scheduling mechanism. Their tests verify that a network node has enough processing power to accept a new connection and to evaluate the minimum delay to be assigned to such a connection.

4.1.9 QoS termination

The QoS termination procedure deals with the proper freeing and updating of system resources as well as killing of processes, once the service has been used. The QoS termination activity is not trivial especially when the communication protocol between the system components is complex or unreliable. In multi-user environments, it is imperative that a shared process is killed only when all sessions are terminated and not just one session.

4.2 QoS Adaptation

It is very common that multimedia systems cannot sustain negotiated levels of QoS during a session for some period of time. This time period can range from a short to an extended period depending on the factors that created the downfall in the QoS. There are several reasons of such factors. For example, the basic architecture of the Internet makes it a best-effort network service making it hard or impossible to reserve resources for connections. Even when QoS is guaranteed, violations in QoS can occur due to network congestion, transient node faults, network reconfiguration etc. QoS adaptation attempts to provide the tools for applications to be more tolerant and to gracefully cope with such unexpected QoS variations.

The adaptive approach has several advantages:

- in some contexts, it might be a necessary pre-condition. For example, in mobile environments, it is hard to maintain QoS guarantees over extended periods of time [Gecsei 97]. This is due to the techniques used in wireless communications usually provide limited data rates and increased error probability in comparison with wired networks.

- From an economic point of view, adaptive applications are resistant to evolutionary changes in the network environment and hence have more lifetimes and increased viability.
- It reduces the probability of QoS violations that may be noticed by the user and thus increases the user confidence in the service provider.
- It can make optimal utilization of the system resources and increases system availability [Bochmann *et al.* 96].
- It can simplify the QoS negotiation process as well as resource management. In some cases, these two activities can be omitted altogether.

4.2.1 Adaptation Mechanisms

There is presently no present methodology to provide adaptation mechanisms in applications. There have however, been ad hoc implementations of adaptation techniques into multimedia applications. Most adaptation mechanisms are pre-defined operations that are statically executed. From these implementations, two forms of adaptation can be distinguished:

- The adaptation procedure does not degrade the initially agreed QoS parameters between the user and the system. Any control action does not have any impact on observed parameters.
- The result of the adaptation procedure is a graceful degradation in the QoS initially negotiated. In such cases, adaptation changes some observed parameters such as the transmission rate.

Adaptation without QoS degradation

Many researchers believe that the main aim of QoS adaptation is to try by all means to maintain the desired quality of service of the user. There have been a number of implementations that have addressed this concern for maintaining the QoS during traffic congestion.

Adaptation mechanisms are usually executed in response to violations observed in QoS parameters such as loss rate and jitter. A number of proposals have been made to effectively reduce the variations in these parameters. Fluctuations in jitter can be eliminated by buffering the received data at the destination end before play-out [Verma *et al.* 91]. This however, introduces an additional delay equal to the amount of jitter that is compensated. It also requires additional buffering space. Data blocks arriving with a jitter larger than the bound that can be compensated will be lost. Loss rate can also be reduced by using a protocol with redundancy.

To adapt to variable data rates, interpolation techniques are commonly used to perform adaptation. Missing data such as video frames and audio samples, can be interpolated from the previous and following segments. The performance of such techniques depends on the media, video content, and the compression techniques used [Ramanathan & Rangan 93].

The Video Datagram Protocol (VDP) [url VDP] is a real-time protocol that permits audio and video presentations to start before the source files are downloaded completely. One of its aims is to prevent the dropping of frames at the receiver side because of late arrival or because the CPU cannot decode the frames in time. VDP achieves its adaptation by continuously monitoring the network throughput and the processing power of the client, and by adjusting the transmission speed at the server side. While the frame rate decreases, the QoS perceived by the client is stable.

[Hafid & Bochmann 97b] propose a scheme whereby the initially agreed QoS is maintained by recovering from any occurrence of a QoS violation. Adaptation occurs at the configuration and component levels. At the configuration level, on violation detection, the system tries to make a transition from the primary overloaded components to alternative non-overloaded components that can offer the initially desired QoS. For example, in a News-on-Demand implementation, a source file on an alternate network is selected if there is congestion on the network where the primary source file resides. At the component level, non-overloaded components reserve additional resources e.g. buffers and CPU slots to provide an improved QoS in order to compensate for violations from other components.

In some situations, protective measures are taken at the resource management level to prevent violations. [Dan *et al.* 95] propose a load balancing mechanism for the dynamic optimization of disk use in an online video server. The server stores a number of videos on multiple disk drives, and each video is divided into a number of fixed-sized segments. A single copy of each video is stored and some spare space is left in the system for creating segment replicas. When a segment is played out for a request, it is also replicated one or more times on different disks so that it is available for subsequent requests. This unloads the original disk.

The Tenet Group at the University of California at Berkeley propose a network adaptation scheme that dynamically manages the provided QoS by changing the parameters that specify it during the lifetime of the connection [Parris *et al.* 93]. Dynamic re-routing, network load balancing and control mechanisms are used to attempt to maintain the initially agreed QoS. Load balancing uses an algorithm defined to determine whether to shift the load on a route. Dynamic re-routing establishes an alternate route and performs a "transparent" transition from the primary to the alternate route.

Adaptation with QoS degradation

Most adaptive applications respond to traffic congestion or violations in QoS parameters, by degrading as gracefully as possible, the desired QoS level. The degradation can be done based on a "degradation path" that the user specifies at the start of a session. The degradation path contains the actions to be performed, usually in a priority fashion, when degradation of the service is to be performed.

A common approach to perform degradation in multimedia communications is to completely suppress the transmission of one media, starting with the one with the least priority. Videoconferencing applications usually prioritize media in the following order: audio, followed by data and video. Hence, the bandwidth is allocated to the audio source first and any remaining bandwidth is shared by data and video [url NetMeeting]. [Hafid *et al.* 97] propose a complete replacement of all real-time sensitive media with simpler media. The idea is to suppress audio and video in cases of congestion with text. It is not clear how such an adaptation can be achieved in a generic manner.

There are variations to the media suppression scheme; instead of completely omitting one media, its QoS level is decreased to a reasonable level [Kanakia *et al.* 93]. For example, instead of transmitting a video with the desired big screen layout, a smaller layout requiring a lower transmission rate, is used. Rate and control mechanisms are present in order to adapt coding parameters and vary output rate [Huard *et al.* 96]. Usually, the user specifies a threshold beyond which the QoS level becomes unacceptable. When such a threshold is reached, the connection is terminated.

A number of methods take advantage of the way video and audio are represented to perform degradation. This has given rise to scalable encoding and dynamic rate shaping techniques, which allow the representation of media at different levels of resolution using multiple bit streams [Wee *et al.* 96]. For example, video is constructed as consisting of a base layer with successive enhancement bit streams. The base layer contains low- resolution information such as luminance, while the enhancement layers contain high- quality information such as chrominance. In cases of congestion, only the base bit stream is sent. Transmission of the MPEG video [url MPEG], which contains prioritized I, B, and P frames often works on the principle mentioned above.

[Yin & Hluchyj 91] propose a similar technique for packet voice adaptation based on embedded coding of voice. A segment of a voice waveform is generally encoded into a block of bits in which several, usually four, sub-blocks are embedded. The sub-blocks are of different levels of importance for reconstruction. Hence, during congestion voice systems usually omit the transmission some of these sub-blocks.

A common technique used for variable data rates is dynamic compression. [Bolot *et al.* 94] describe a scheme which reacts to data rate changes by dynamically changing the coding parameters such as the compression ratio. This requires a flexible coding scheme that operates over a range of compression ratios, and a protocol between the source and the system. The protocol is usually needed to compute the available bandwidth and to estimate the quality of reception at the sinks.

4.2.2 Adaptation Frameworks

In this section, we describe three cases of QoS adaptation as proposed and implemented in different contexts. The techniques presented in these frameworks give a fair insight of the mechanisms usually employed in adaptive systems in general.

Distributed Multimedia Applications QoS Adaptation architecture

Proponents of this architecture treat QoS adaptation related to a case study on a News-on-Demand distributed multimedia application [Hafid & Bochmann 97b]. An adaptation approach which allows to recover automatically, if possible, from QoS violations by:

- identifying a new configuration of system components (such as a video display, decoder, network or video server) which might support the initially agreed QoS and by performing a user-transparent transition from the initial configuration to the new one.
- redistributing the levels of QoS that should be supported in the future by the components.

- redistributing the levels of QoS that should be supported immediately to meet end-to-end requirements. This is based on the principle that local QoS violations at one component, may be recovered immediately by other components supporting the same service.

The mechanisms assume that the distributed system supports QoS guarantees. Components are thus able to reserve resources to support certain levels of QoS.

Adaptation is performed at the configuration level based on a Component Reconfiguration Scheme (CRS). When a violation is detected, one or more alternate components are selected and a transparent transition from the primary components to the alternative ones is performed. The alternative components are selected based on several factors such as the functional configuration of the requested service and the current load of the system components. This approach is suitable for applications such as video on demand that require QoS guarantees.

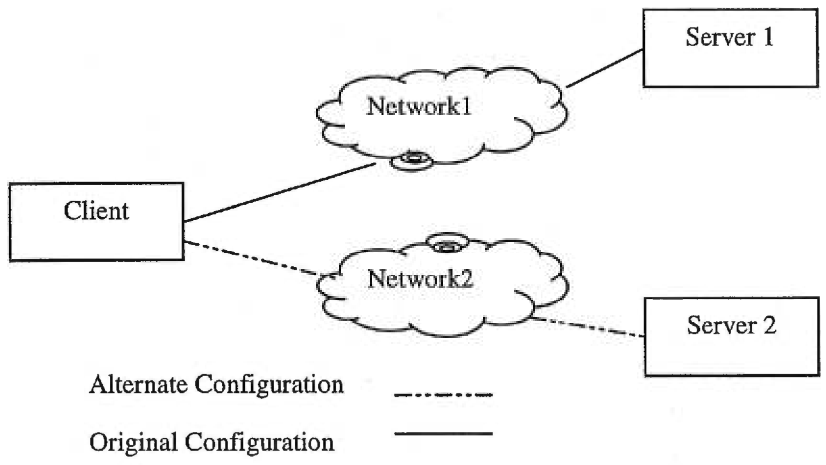


Figure 4.1: Adaptation at the configuration level

As illustrated in Figure 4.1, in the News-on-Demand application, if the primary file server cannot meet the agreed quality of service, another source file server on an alternative network that can meet the required QoS, can be selected.

Adaptation is also performed at the component level in the Resource Reconfiguration Scheme (RRS). When a component violates its guarantees, some form of cooperation among components is started with the aim of reassigning the guarantees to the different components, such that end to end guarantees are met. Thus, the non-overloaded components may reserve additional resources e.g. buffers, CPU slots to compensate the violations of other components, by providing an improved QoS. This approach is appropriate for teleconferencing applications which are have stringent temporal requirements.

For example, as illustrated in the Figure 4.2, let us say that the configuration for an audio conference between two clients are Client1, Network1, Gateway, Network2 and Client2. Each of these components commits to a certain level of QoS. During the communication session, Network1 fails to meet the agreed delay while Network2 has a some unused amount of resources. Under RRS, Network1 can ask Network2 to reserve more resources to compensate for the violation at Network1. Later, if Network1 has enough resources to meet the initially agreed QoS, it can notify Network2 to free the excess resources used.

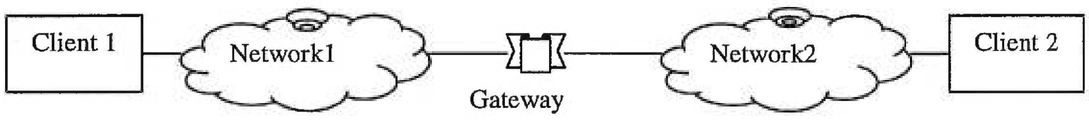


Figure 4.2: Adaptation at the component level

The framework consists of a QoS Manager and a number of QoS agents with each agent representing a component. The QoS Manager manages the system components in order to provide QoS guarantees while the QoS agents collect performance monitoring information about the components they represent. While scientists in the artificial intelligence area usually refer to an "agent" as an entity that has properties such as autonomy, persistence, communication, mobility and reasoning, the concept of an agent in this context is quite different. The notion of agent was applied to refer to the monitoring and calculation performed by these components and the ensuing communication with the QoS Manager.

The authors propose a third scheme called the Delay Recovery Scheme (DRS) The idea behind DRS is the same as RRS except that it tries to compensate for delay violation for each transmitted data unit. If the measured delay value of a data unit is below the commitment, a VIOLATION signal is sent with the data unit in question. The next component in the chain may apply a higher QoS if possible to that data unit. The existence of synchronized clocks is assumed for easily determining the delay of each data unit with the help of timestamps.

The proposed schemes support a graceful degradation. If an alternate configuration to support the initially agreed QoS cannot be found, a configuration supporting a lower QoS can be automatically selected in a graceful way; e.g., playing black and white video instead of the initial color video. The authors specify that a scheme that supports the characteristics of RDS, RRS and CRS may be more efficient than just using one scheme alone.

QoS Adaptive Transport system

Proponents of this framework from the University of Lancaster, describe the implementation of a QoS adaptive transport system which incorporates a QoS oriented API and a range of adaptation mechanisms in order to adapt to fluctuations in QoS [Campbell & Coulson 96]. The work is based on the Multimedia Enhanced Transport System (METS) which supports multi-layer coded flows in a multicast, multimedia networking environment in which client workstations have varying capabilities [Campbell *et al.* 93].

QoS is specified at the API level, which is a set of extensions to the Berkeley socket API, in terms of a flow specification and a QoS policy. The flow specification includes parameters such as jitter, throughput, delay etc. The QoS policy allows the infrastructure on how to deal with the flow when resource availability changes.

QoS adaptation is provided by a flow management plane mechanism designed to exploit the layer encoding property of media formats such as MPEG. MPEG structures its video streams into 3 layers: a base layer with successive enhancement layers. For example, for MPEG1, the base layer is represented by I pictures and the enhancement layers by the P and B pictures respectively. The coding of B-pictures depends on the previous and following I-pictures or P-pictures. The coding of MPEG streams is shown in Figure 4.3. Discarding a B-picture results in the loss of only one picture frame, while losing an I-picture results in being unable to decode 2 B-pictures in an adjacent Group of pictures (GOP), and all the pictures present in the GOP.

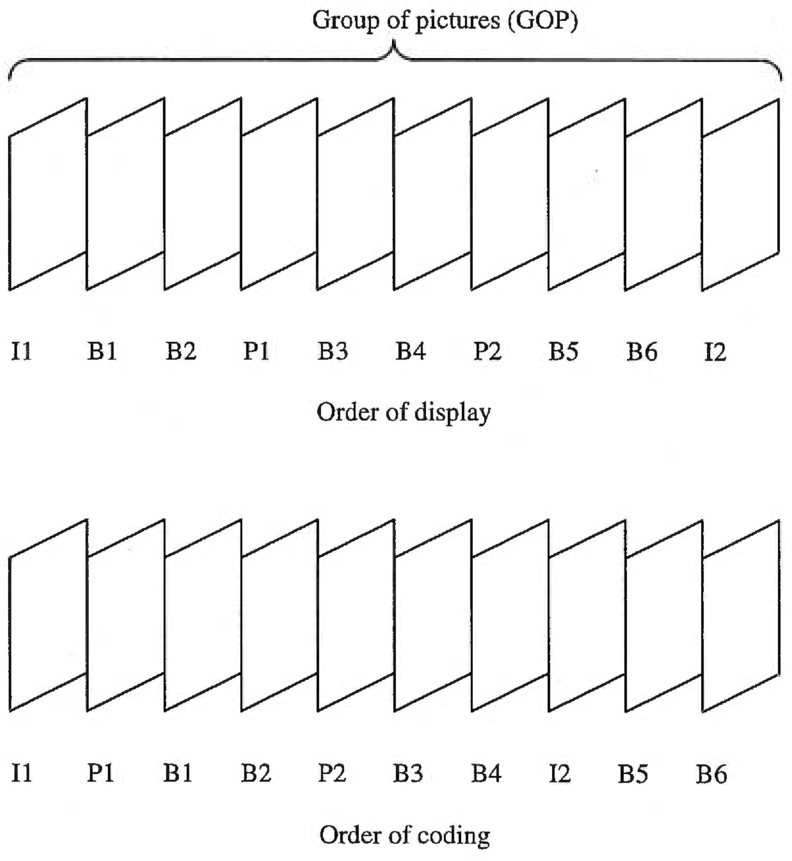


Figure 4.3: MPEG frame coding [Cho & Shin 99]

QoS adapters scale flows, based on the user supplied QoS policy, by adding or removing enhancement layers or instantiating filters on QoS fluctuations. The term

scaling is used to refer to the dynamic manipulation of media flows as they pass through a communications channel.
Instances of the QoS adapter are present in both end systems and network switches. The QoS adapter periodically probes the network to test for resources that have just become available or unavailable. At the start of a cycle, called an era, the local QoS adapter instance of each receiver determines its local resource availability and sends a RES signaling message containing an indication of the additional bandwidth required to support the addition of enhancement layers. RES messages are always forwarded towards the core switch in the network. When the adaptation protocol notes that congestion has passed, it increases the bandwidth demands accordingly. Congestion can be noted when switches detect that queues are building beyond a pre-defined threshold.

The adaptation protocol supports the concept of end-to-end adaptivity and takes into account end-system resource availability. In order to work on multicast environments, the QoS adaptation protocol provides support for the merging and forwarding of RES messages. It does this by building periodic merged messages based on an interval time before forwarding to the core switch.

One disadvantage of the currently implemented scheme is that discrete fluctuations may be undesirable in certain application contexts. A solution to this problem is the notion of continuous adaptation using the dynamic rate shaping filter by means of source bit rate filters. These permit fully continuous adaptation of bit rates.

RTP Feedback-based Dynamic QoS Control

The proponents of this adaptation scheme propose mechanisms for the dynamic adjustment of bandwidth requirements of multimedia applications as a means to adapt to congestion situations [Busse *et al.* 95]. More specifically, the mechanisms use a feedback mechanism based on the Real Time Transport Protocol (RTP) to control the bandwidth of an videoconferencing application according to the network load.

RTP allows receiving end applications deliver RTCP receiver reports to the source application. The source analyses these reports and computes statistics about the the packet loss of all receivers, packet delay jitter, as well as the round trip time to the receivers. During the RTCP analysis of the packet loss rate, a low-pass filter is used to smooth the statistics and to avoid QoS oscillations. The source then determines the congestion state for each receiver. The smoothed loss rate is computed by a low-pass filter.

Three categories of congestion are defined namely: unloaded, loaded and congested as shown in the Figure 4.4. The upper threshold λ_c is chosen so that the loss rate is tolerable by the application, and the lower threshold λ_u is chosen low enough so as to avoid QoS oscillations.

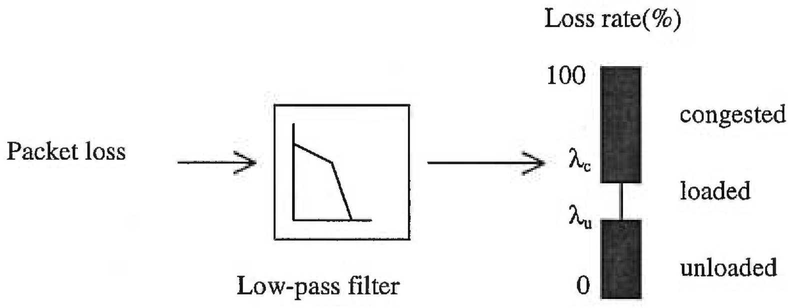


Figure 4.4: Receiver classification

Based on the classification of network congestion, a decision on the bandwidth adjustment is made. When the application works in a unicast mode, the network congestion state can be directly mapped to either decrease, hold or increase. In a multicast environment, the authors propose two algorithms; the first one takes into account the receiver with the highest average loss rate. Based on the classification of this receiver, the bandwidth will be adjusted accordingly. While in this case, congestion will be avoided, the drawback is that receivers on high-bandwidth links might be unnecessarily hampered by those on low bit-rate connections. The second algorithm takes a proportion of receivers in either unloaded, loaded and congested states. Based on the proportion of congested users on a certain threshold, the bandwidth is decreased, held or increased. The bandwidth adjustment is based on a well-known scheme used in TCP and ATM rate-control ABR traffic. In cases of congestion, it uses a multiplicative increase. On the other hand, the decision to increase bandwidth only leads to an additive increase.

Implementation of the adaptation mechanisms were performed on the vic videoconferencing application [McCanne & Jacobson 95] which is widely used for MBONE transmissions of conferences and workshops [Eriksson 94].

Like all RTP implementations, there is a need to balance the desire for up-to-date control information and the desire to limit control traffic to a small percentage of data traffic. In sessions consisting of a considerable number of users, if the number of feedback RTCP packets is not controlled, this can lead to serious scaling problems.

In this chapter, we have presented several key principles of QoS management, and we focused mainly on QoS adaptation which is a central part of our proposed architecture. In the next chapter, we describe in detail our proposed architecture. We show how the H.323 standard and techniques such as adaptation are used to provide QoS.

Chapter 5

A Proposed QoS Architecture

5.1 Goals and Objectives

The motivation for our work was to build a prototype tool for a real-time videoconferencing application using on-line training as case-study, and designed to provide the mechanisms for satisfying user QoS.

Tele-teaching applications need to provide the basis for a workgroup structure and rules that govern the interaction among participants. In parallel, there is the media manipulation that treats audio, video and data streams, which needs to be considered. The RTP protocol as described in Chapter 2, takes care of this manipulation. QoS management is also crucial to maintain the integrity and quality of the media communication among all participants. We particularly focus on adaptation of QoS and implement strategies that control the adaptive behavior of the application based on observed measurements. Adaptation techniques based on some techniques described in Chapter 3 and 4 are described and implemented. Finally, one of our goals is to produce a tool that can communicate with other tools or products already on the market. We designed our application based on industry standards so as to make it interoperable and compatible with other standard-based products. We made use of the H.323 standard for that matter.

5.2 Functional Overview

Figure 5.1 illustrates an overview of the system. A modular approach was followed

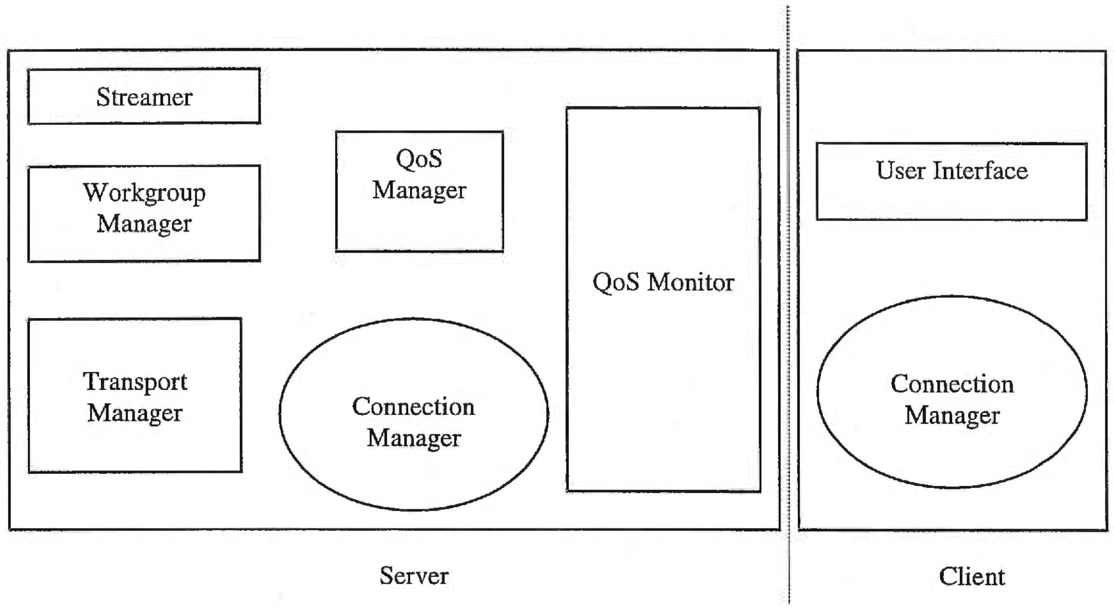


Figure 5.1: System overview

for simplicity reasons and also to clarify the tasks performed by each component. The system is divided into two modules: the server and the client.

On the server side, the *Streamer* component is responsible of the generation of streamed media such as audio and video. The *Workgroup Manager* performs workgroup management as part of cooperative activities among participants. The role of the *QoS Manager* is to perform QoS management activities, including adaptation, and call admission control for new connections. The *QoS Monitor* component has been exempted from the *QoS Manager* process and provides monitoring facilities. The *Connection Manager* takes care of connection setup and tear down between clients and the server. The *Transport Manager* is responsible for managing the multiple network connections between the server and the clients. We explain these multiple connections in the next section.

On the client side, requests for a connection setup or tear down to the server are made through the *Connection Manager*. A user interface component also exists at the client side for the user to specify his/her QoS requirements. We note that instances of the client module can be generated and replicated in case of multipoint or multicast sessions.

5.3 Multiple Network connections

With the decreasing low-cost connection rates, it is becoming more and more common to see users being connected to different service providers. We extend this paradigm futher in our design, and propose a scheme whereby a user is connected to the same service provider but through two or more networks. This idea is in accordance to the advent of the new Internet favoring different classes of service. Figure 5.2 illustrates the idea of multiple network connections.

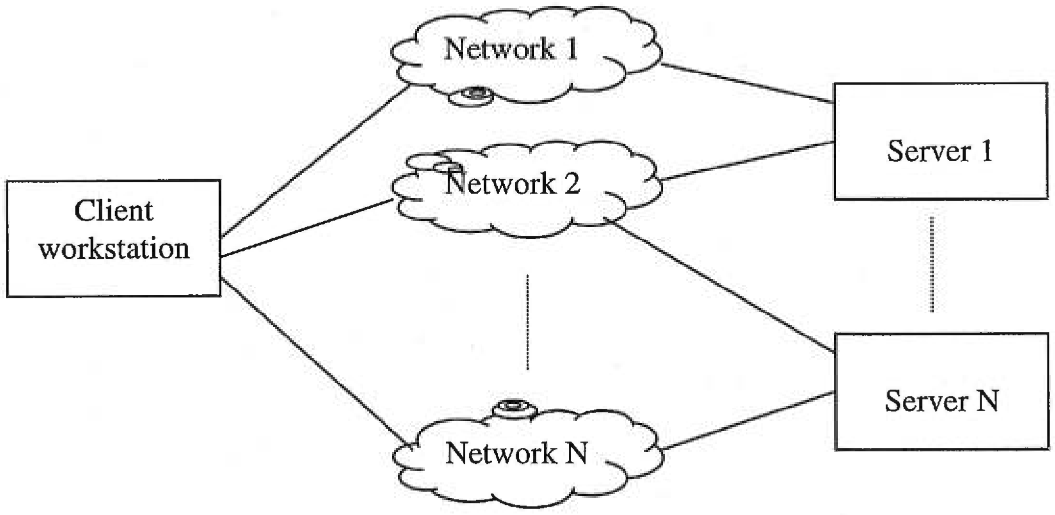


Figure 5.2: Multiple network connections

These different networks might be similar in nature. For example, the networks to which the user is connected, might be running on the TCP/IP protocol. Each network however, provides a different quality of service, with one offering a premium service while the other one offering ordinary service. The difference in service provided might be provided through the use of a faster backbone such as ATM or by restricting the number of connections on the premium network [Romanow & Floyd 95].

The networks can use different technologies also. For example, the user can be connected through the Internet and using TCP/IP on one network. At the same time, he/she could be connected to the same service provider on an ATM network using native-mode ATM as protocol.

We can extend the concept of multiple network connections to other network technologies such as ADSL, wireless etc. The architecture we propose is generic so that the transport protocol implemented is independent of the network layer underneath.

5.3.1 Reasons for multiple network connections

There are several reasons that might favor such a configuration namely:

- Several variants of multimedia documents may be stored on different file servers, which are connected to a client's workstation via different networks. For example, video and audio can be streamed from one server while data is being transmitted from the other server. Synchronizing media at the client side becomes an important issue.
- In cases of congestion on one network, the availability of another connected network gives the possibility of shifting some of the load from the congested network. The goal might be to maintain the user QoS desired level or to increase it by shifting the streams on the less-congested network.

5.3.2 Cost Issues

The cost factor is a key issue in the choice for a multiple network configuration. Obviously, if there were no cost factor, users would be choosing the fastest network with the highest QoS. It is up to the user to decide whether it is necessary or cost effective to have different network selections. The choice might be influenced by the type of application used e.g., whether real-time applications are used, the monetary amount involved, and the services being provided.

5.3.3 Generic Design

The architecture we propose is generic so that the transport protocol implemented is independent of the network layer underneath. For explanatory purpose, and in the remainder of this thesis, we shall consider two network connections which we shall refer to an IP network and an ATM network. The IP network is the primary network and secondary network is the native mode ATM network using AAL5 as transport layer. We assume obviously, that the ATM network offers faster and better service than the IP network.

5.4 The General Architecture

The architecture in figure 5.1 shows the interactions among all the components in the system and the suite of protocols used.

The grey boxes are the components described in the previous section that provide the functional requirements of our prototype tool. We worked on this architectural design so that the tool could comply by the standards defined by ITU's H.323 videoconferencing standard and hence ensure its interoperability with other H.323-compliant products.

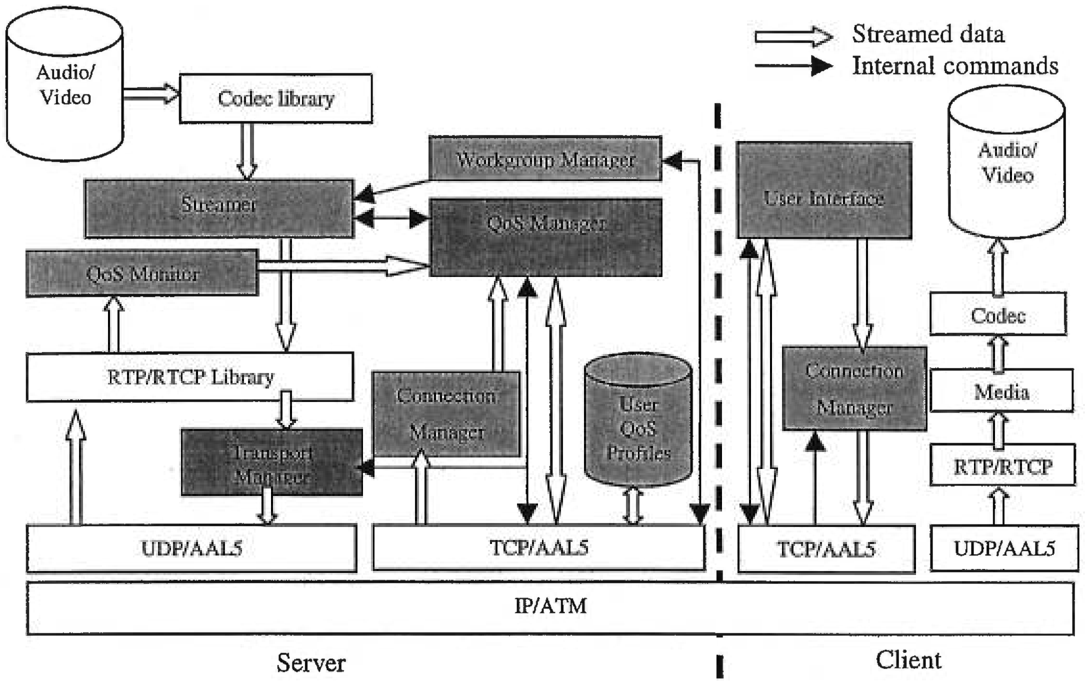


Figure 5.3: General architecture

5.4.1 Terminology

In presenting our proposed architecture, we will be referring to the following common terminlogy which we explain in finer details.

A media connection

There are two approaches that are used to transmit related streams e.g., the video and audio streams of an application. They are the single channel approach and the multiple channel approach.

In the single channel approach, both video and audio are transmitted using a single channel. This helps preserve the temporal relationships i.e., synchronization, between the media. One of the drawbacks in using this approach is that if the channel is interrupted due to some problem, no media can be delivered to the destination. Also, usually video and audio can have different QoS requirements which have to be specified separately.

The multiple channel approach takes care of these problems. In this approach, each media is transmitted via a separate channel and each channel supports the QoS required by the transmitted media. We refer to each channel that is established for one media as a media connection. Synchronization mechanisms are required to restitute the temporal relationships between the different media, and we shall refer to that issue in a later section.

QoS violations

A QoS violation is determined by the *QoS Manager*. It occurs per media connection i.e., there can be a QoS violation for an audio connection and another one for a video connection. A QoS violation signal is generated whenever the observed parameters are "below" the expected parameters; e.g., the observed packet delay is lower than the expected packet delay.

We base our measurements for QoS violations on packet delay and packet loss. We omit throughput in that calculation because it is common to find videoconferencing situations where throughput levels exhibit high variations due to increased video movements.

5.4.2 The Streamer

The *Streamer* component is responsible for media acquisition and packetization. The acquisition can be made through a file, a network port or, more commonly via an input device such as a camera. The media information is then tokenized and packetized into RTP packets.

For each media connection created, the *Streamer* instantiates an object called MMSession which serves as a stream manager for that connection. The following figure illustrates the *Streamer* component.

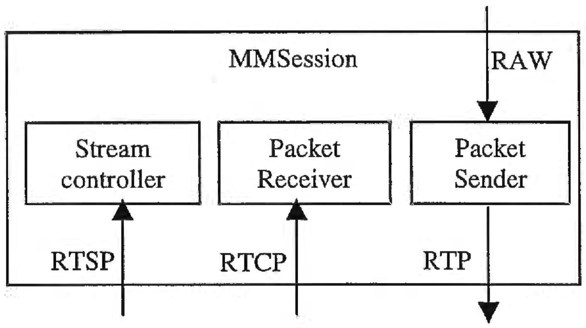


Figure 5.4: Streamer component

The MMSession object contains three sub-objects namely; the stream controller, the packet receiver and the packet sender. The stream controller provides stream manipulation functionality such as START, STOP or PAUSE audio or video streams. It is based on the Real Time Streaming Protocol (RTSP) which is a standard for streaming video on the Internet [Schulzrinne *et al.* 98].

The media information is carried in RTP packets, which are encapsulated into UDP packets. The packet receiver object receives control information in the form of RTCP packets. Such feedback information from participants allows the *Streamer* to analyze the status of the network and connections and hence make any adjustments to the streams as needed.

5.4.3 The Workgroup Manager

The role of the *Workgroup Manager* is to offer the services that allow for cooperative work among participants. The services provided by the *Workgroup Manager* are:

- creation of groups and subgroups.
- floor control principles to maintain the integrity of the cooperative work and temporal synchronization among participants.

Groups and subgroups

In making a connection request to the server, each client needs to adhere to a group. The *Workgroup Manager* at the server level is responsible for the creation and deletion of groups. Each group created is assigned a Group ID to help locate connections that are meant for the same group. Subgroups can also be created to allow for collaborative work among teams dealing on specific subjects within a group.

Floor control mechanisms

The *Workgroup Manager* performs routine tasks such as managing the entries and exits of members in and out of a group in order to keep the group in a coherent state. We came up with the following rules governing the entry into groups and group streaming namely:

- the number of participants in a group is only limited by the number of connections that the application can provide. Therefore, a connection request to join a group is dealt with by the *QoS Manager* as part of the call admission control process.
- During on-going transmission of media streams in a group such as streaming of a video clip, newcomers to that group will receive streaming at the current point of transmission.
- In parallel to the point above, only one client member can start group streaming. By default, the first member to enter a group is granted that special permission. We refer to that member as the "Chief Member". The chief member can also control the stream through commands such as PAUSE, RESUME, PLAY etc.

5.4.4 The QoS Monitor

The role of the *QoS Monitor* is to provide measurements of the network status. It uses the RTP protocol as a means to compute QoS indicators such as current network throughput, delay jitter, packet loss, and packet delay. It does so for each media connection. For every two hundred RTP packets that it generates towards a connection, an RTCP packet is returned from the client. This amounts to at least four receiver reports being received per media connection every second. The *QoS Monitor* compiles these reports and calculates the QoS indicators before sending them to the *QoS Manager* for analysis. The *QoS Monitor* does not have any notion of QoS violation; its role is mainly to calculate the parameters and send them to the *QoS Manager*.

5.4.5 The QoS Manager

The *QoS Manager* provides the following four principal functions:

- It performs call admission control to determine whether to accept new connections or not.
- It maintains a user profile of clients to determine their preferred requirements.
- It determines whether there are violations in the specified QoS level.
- It provides adaptation strategies in response to QoS violations of a user connection, based on the latter's user profile. It also provides the adaptation strategies for upgrading the QoS towards the desired level following a period of congestion.

Besides these functions, the *QoS Manager* handles the various tasks of QoS management such as QoS mapping, negotiation and termination.

QoS mapping

Since our tool is designed for the "best effort" service offered by the Internet, there is no mechanism to reserve resources at the network level. QoS levels are mapped directly on bandwidth, delay, jitter and loss rate values. This means that there is a set of throughput, delay, jitter and loss rate values that is calculated for each level of QoS. Such mapping depends mainly upon the codec being used [Wakeman 93]. The

H.323 standard codecs of our application gives us a fair indication of the calculation of these value indicators.

Throughput mapping

Since the basis of the application is dependent upon RTP packet feedback, an approximation of the RTP packet throughput required for a certain user desired QoS level is computed. We come up with the following formula for the required throughput before compression: For video,

$$\textit{Required Throughput} = \textit{Frame Size} * \textit{Number of Colors} * \textit{Frame rate}$$

For audio, the throughput required is specified for each sampling rate of the codec used. If we take into account the compression rate as specified by the codec and we add the RTP header, we can come up with an approximation of the required RTP packet throughput.

Delay mapping

We came up with the following approximations of delay indicators. For audio and video, a delay inferior to 200 ms is commonly referred to as being tolerable. Any delay exceeding that threshold leads to a QoS violation.

Loss rate mapping

There are two reasons for packet loss. Packets get lost due to buffer overflow or due to bit errors. The probability of bit errors is very low in most networks, and most losses are due to congestion resulting in buffer overflows. The loss rate was given a fixed value of 5% for all QoS levels. This value was estimated based upon experimental results.

Call Admission Control

The *QoS Manager* is the component that decides whether a new connection can be met by the system, i.e., whether the desired quality of a new connection can be met within the constraints imposed by the system. Such an activity is commonly referred to as call admission control. It takes the bandwidth and the number of available connections as the indicators for accepting or rejecting a connection.

Through experimental results, we gave an arbitrary number of twenty as the maximum number of connections to be allowed concurrently on the system. Once this number is attained, all new requests for connections are rejected. This value was reached after several experiments of the system; beyond twenty connections, the quality of the videoconference deteriorates to a less than tolerable level.

If the maximum number of connections has not been attained, an estimate of the bandwidth between the client and the server is then performed, to see if the bandwidth requirements can be satisfied. While during the flow of a connection, the link bandwidth between the client and server is reported by RTP reports, no such facility is available prior to connection establishment. We devised a lightweight "ping" call to retrieve the bandwidth information. Such a procedure needs to be fairly quick because it affects the connection establishment time. If the calculated bandwidth is found to meet the required bandwidth, the *QoS Manager* accepts the connection request orelse it rejects it and specifies the reason of the denial.

The user profile

A user profile, also known as a degradation path, is maintained at the server. It contains a series of recommendations by the user, specifying the steps to be followed in case the desired QoS cannot be maintained during the life of the connection. The following information is maintained in the profile:

- ➢ The priority of media. This specifies which of audio or video is the priority media. In case of QoS degradations, the QoS level of the media the user considers less important will be decreased first. In situations of QoS upgrades, the QoS level of the more important media will be increased first.
- ➢ The QoS levels the user deems desired and tolerable for each media. If the tolerable thresholds cannot be maintained on the network used, the connection on that network is automatically closed.
- ➢ The choice of another network. In case QoS degrades below acceptable levels on the current primary network, the user specifies whether routing media on a secondary faster network is possible. This entails higher cost issues that the user is willing to incur.

Each user having a connection must have a permanent or temporary user profile at the server. Users have the option of saving their profile on the server for future use in order to avoid repeating the process of setting up the user profile again.

Adaptation strategies

The recommendations stored in each user profile determine the adaptation strategies that the server adopts for that particular user, whenever the QoS offered by the system is not the user desired level. There are two possible cases for the *QoS Manager* to invoke an adaptation strategy namely:

- ➢ In response to a QoS violation of a media connection after analysis of reports obtained from the *QoS Monitor*.
- ➢ After a time period in which no QoS violation has been reported and if at that moment, the current QoS level is inferior to the desired QoS level. The adaptation mechanism is executed to upgrade the current QoS towards the desired QoS level.

Adaptation in response to QoS violations

In response to a QoS violation signal, the *QoS Manager* consults the user profile of that user to determine what strategy to adopt. The three essential information for it to come up with a decision are the media priority, the QoS levels desired and tolerable to the user, and the possibility of using a faster network at a particular cost. It then performs the following steps.

1. Decrease by one level, the QoS of the media with the least priority.
2. Compare the current network throughput with the throughput needed to transmit at new QoS level configuration.
3. If the current network throughput is greater, then maintain this QoS level. If it is lower, then repeat step 1.
4. If the threshold QoS level of the media is reached after degradation, then repeat step 1 with the other media.
5. If all thresholds have been reached,
   a. And there is no option of using a second network, the connection of the user is closed.
   b. And a faster network is a possible alternative, all media connections for that user are shifted onto the faster network, and the QoS provided is the desired QoS level. Repeat step 1 in case of QoS violations on the faster network.

Adaptation in response to non-violations

As we have seen in the previous section, it happens during the lifetime of a videoconferencing session that some network congestion causes the system to degrade the QoS or even switch to the faster network. The congestion might be temporary. In such cases, mechanisms are needed to upgrade the current QoS towards the desired level. We devise two strategies for upgrading QoS namely:

- ➢ If the current network of transmission is the same as the primary network, and the present QoS level is inferior to the desired QoS level, and that three consecutive RTCP reports generating no QoS violations are obtained at the server level, the current QoS of the highest priority media is increased by one level. This step repeats itself for every three reports until the desired QoS level is attained.
- ➢ If connection has switched from the initial to the faster network, the adaptation mechanism will attempt to switch back to the primary network if it receives nine consecutive RTCP reports generating no QoS violations. Once it switches to the initial network, the desired QoS levels are restored. The motivation to switch to the primary network is mainly a cost issue and policy. It costs less to connect on the slower network.

We note that the algorithm to upgrade QoS is more conservative than when to decrease QoS. This design decision is based on the fact that best-effort networks are more liable to be traffic congestion leading to degradations in QoS rather than an upgrade in the provided service. In Figure 5.5 and Figure 5.6, we outline the algorithms of the QoS degradation and upgrading processes respectively. Both algorithms have linear complexity. We use pseudo-code methodology for a clear understanding of the algorithms.

```
If QoSViolation()                  //QoS Violation detected
    p = lowest;                  //Start decreasing least priority media
    Degrade = True;              //To degrade QoS level
    media = MediaPriority(p); //Select media to decrease
    While (Degrade)
        If (QoS(media) = threshold(media))      //If threshold of media attained
            Media = MediaPriority(p+1); //Consider higher priority media
            If (Media = NoMore)          //If all media have been considered
                If (CurrentNetwork = Primary) & (SecondaryNetwork = True)
                                //Possibility of second network
                    For (AllUserMediaConnections)
                        SwitchNetwork(AllConnections)
                            //Switch all user connections to secondary network
                        QoS(media) = UserDesiredQoS(media)
                    EndFor
                    Exit();
                Else
                    CloseConnection() //Close connection
        Else
            QoS(media) = QoS(media) – 1;
                //Decrease QoS of media by one level
            If (CurrentThroughput > NeededThroughput)
            //If current network throughput can accommodate required throughput
                Exit()
    EndWhile
```

Figure 5.5: Degradation of QoS

```
If NoQoSViolation()                    //No QoS Violation detected
        If (CurrentNetwork = Secondary) & (FavourableReports = 9)
         //If on secondary network and reception of 9 consecutive good reports
                For (AllUserMediaConnections)
                        SwitchNetwork(AllConnections)
                          //Switch all user connections back to primary network
                        QoS(media) = UserDesiredQoS(media)
                        //Provide the desired QoS level
                EndFor

        If (CurrentNetwork = Primary) & (FavourableReports = 3)
         //If on primary network and reception of 3 consecutive good reports
                QoS(media) = QoS(media) + 1
//Start incrementing the QoS of media by 1 level, starting from the highest-priority
media
```

Figure 5.6: Upgrading QoS

5.4.6 The Connection Manager

The *Connection Manager* is responsible for accepting client connection requests. Such requests contain user QoS specifications such as bit rates, connection port numbers and media parameters such as codecs. The *Connection Manager* ensures that the two entities involved in a media exchange agree on the parameters to be used during the connection.

In the H.323 standard, connection management is provided by the H.225 and H.245 protocols. H.225 provides registration, admission control and procedures of disengagement between endpoints and gatekeepers. The H.245 protocol provides the call control mechanism for connection and communication among H.323 terminals. It enables codec selection and other capability negotiations such as bit rate, frame rate and video format. The combination of the H.225 and H.245 protocols is a very powerful yet somehow complex specification to implement. This led us to create a simplified version for connection management that allowed us to introduce application specific mnemonic exchange.

The new connection protocol contains three control messages for connection establishment namely: a request (REQ), an acknowledgement (ACK) and a denial (DEN). The client issues a request message to the server on a predefined port. Following the request, the server answers with either an acknowledgement or a denial message. An acknowledgement specifies that the request has been approved and it contains the necessary parameters such as server port numbers to complete the call establishment. Once the client receives the acknowledgement message, media streaming can start. A denial message means that the connection request has been refused by the server. The message includes the reason for the connection refusal.

The formats for the three messages are illustrated in the following figure.

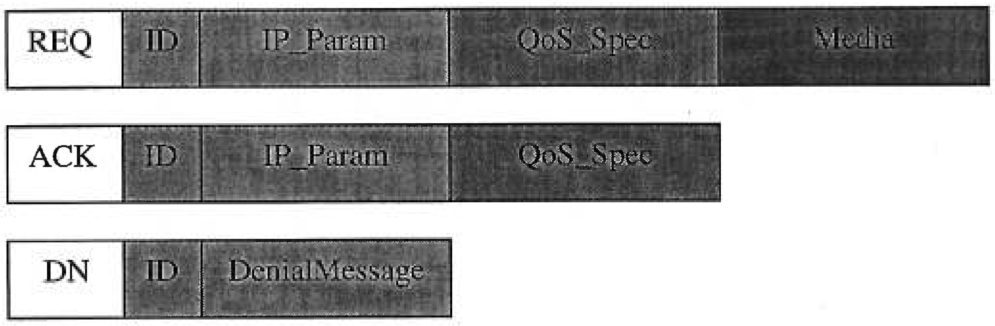


Figure 5.7: Connection message formats

Each of the three packets contains a header identifying the type of packet: REQ, ACK or DN.

The ID field enables the marking of call establishment. Each ID is unique and enables asynchronous replies from the server. This means that the order of several requests to and acknowledgements or denials from the server can be different, depending on the requirements and link of the user.

The IP_Param field contains the necessary information for connection establishment such as client IP address, TCP/UDP ports for control and media exchange, and the session name.

In the REQ packet, the QoS_Spec field represents the desired user QoS preferences whereas in the ACK packet, they represents the capabilities of the server. QoS_Spec is defined as follows:

```
QoS_Spec{
        QoS_VD; //Video Desired Level
        QoS_VT; // Video Threshold Level
        QoS_AD; // Audio Desired Level
        QoS_AT; //Audio Threshold Level
        QoS_P;    //Media Priority
}
```

Finally, the media field in the request packet contains information about the requested media such as the type of media and its sampling rate.

5.4.7 The Transport Manager

The role of the *Transport Manager* is to provide a set of mechanisms whose tasks is to provide treatment to the media streams in correspondence to the QoS media requirements of the user. It manages all these multiple network connections that are possible at the server configuration level. Its principal tasks are to:

- Receive packets for all client connections, and classify them according to the media type and network destination. The sub-component packet classifier performs this task.

- Provide a scheduling mechanism to meet QoS per media, before sending the classified packets on the network. This task is performed by two concurrent packet schedulers.

The following figure illustrates how the *Transport Manager* interacts with other system components to achieve its tasks. It is represented by the grey box.

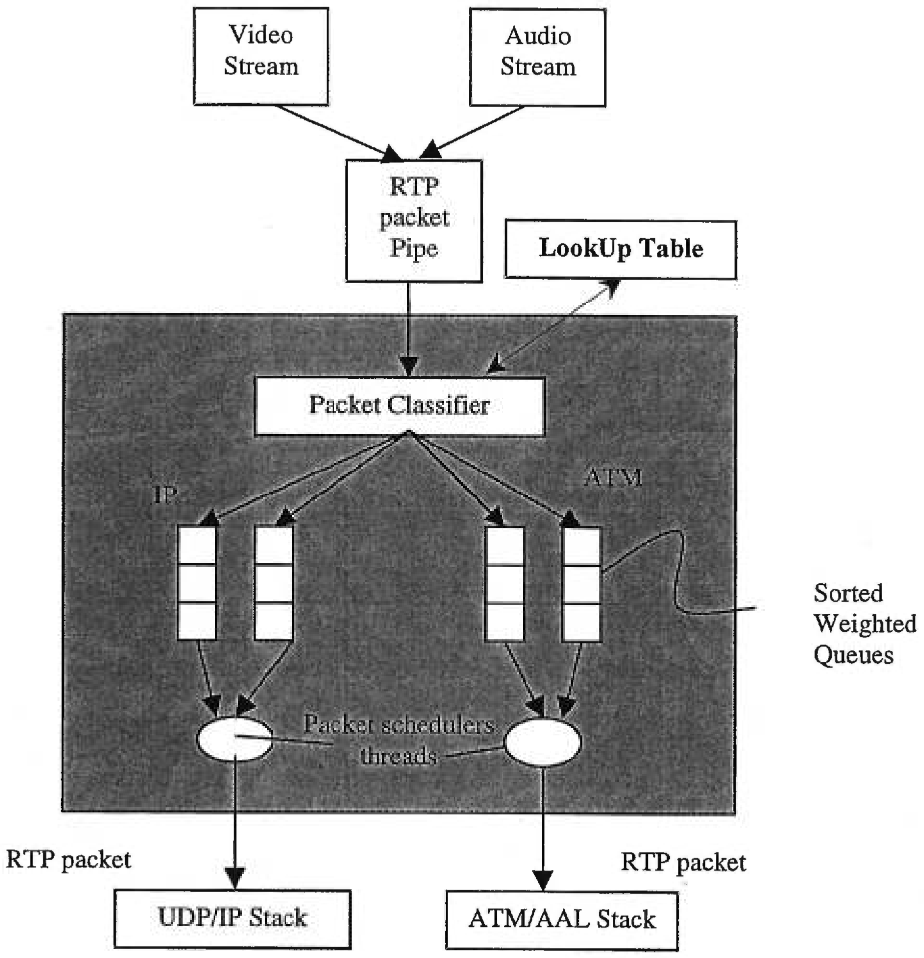


Figure 5.8: Transport Manager

The packet classifier

Video and audio streams are encapsulated into RTP packets and tunneled through a pipe by the *Streamer* component. The packet classifier reads these packets sequentially from the pipe.

One of the tasks of the *Transport Manager* is to consider the quality of service for each separate media specified by the user. Its role is dynamic because it does not treat video and audio equally unless the user specifies so. This concept is in accordance with videoconferencing situations where audio and video have different QoS requirements. The *Transport Manager* uses a queuing and scheduling scheme to achieve this purpose.

The role of the packet classifier is to place the packets in the appropriate queues. It reads out the field "ConnectionID" in each packet and looks for that id in a lookup table, which we reference as the "LookUp Table". The LookUp Table tells the packet classifier to which appropriate queue to route the packet.

"ConnectionID" is a key field that indicates the number assigned to a connection. It is unique for each media connection and generated by the *Connection Manager* component on connection establishment.

The Queues

The queuing system used separates media by type and network destination. The possible media types we consider are audio and video. The network destinations determined by the adaptation algorithm are the IP and ATM network.

In such a scenario, four queues devised; one for audio over IP, one for video over IP, one for audio over ATM, and one video over ATM. The number of queues that the *Transport Manager* supports depends on the different networks that are supported and the number of media to be streamed. Basically, if the number of queues needed is N_Q, the number of media to be differentiated is N_M, and the number of networks supported is N_N, then

$$N_Q = N_M * N_N$$

Each of the queues operate independently on a first-come first-serve basis; an RTP packet entering a queue will be sent on the network prior to any other packet having arrived at a later time in the same queue. However, this FIFO principle loses its significance when all queues are serviced altogether. A packet entering one queue might be sent on the network later than another packet on another queue, even if the latter packet was queued at a later time. It is the role of the scheduling algorithm to service the queues which determines the rate at which packets from each queue will be paced on the network.

The schedulers

Usually, the decision on the prioritization is application-specific, with most applications giving audio priority over video. The schedulers provide a dynamic priority scheme in which priority given to media before sending on the network can change during the lifetime of any connection.

In this scheme, prioritization depends on the percentage of total throughput taken by each media. For each network, the scheduler computes the throughput required by each media; i.e., the required throughput for all active audio and video connections are summed up separately. The ratio is expected to change over time, depending on the state of connections required by prospective clients. For example, newcomers might ask for video connections only, while others might ask for both audio or video connections.

The scheduling applied is based on the weighted-fair queuing principle, where the weights assigned to the queues depend on the respective throughput of the media connections for each network destination. The queues are serviced in a round-robin fashion starting from the audio queue.

As an example, if on the IP network, all active audio connections amount to a bandwidth of 50 KB/sec and all active video connections amount to 100 KB/sec, the video queue is assigned a weight of 2 compared to a weight of 1 to the audio queue.

This means that for each audio packet that is sent on the IP network, two video packets are sent in the same pass. For each network supported, there is a separate independent thread that does the scheduling to ensure a high degree of parallelism.

The reasoning behind the weights assignments of the queues is dependent upon the fact that the time to transmit an RTP packet in any queue is the same. This is, indeed, the case because all RTP packets have a fixed size from the time of packetization to encapsulation into UDP packets.

Besides weighted-fair queuing, there are other new emerging scheduling algorithms that are often applied at the link level. Examples of these are the Deadline-First, Maximum Laxity and the Virtual Clock principles [Zhang & Ferrari 94]. Those algorithms were not considered in this case because they analyze deadline information in the packets and work with timestamps of packets inside the queues. In practice, the complexity of such algorithms is usually maintained at the link level, i.e., at the switch level. Such techniques can be used for comparison with our lightweight scheduling mechanism in a future area of research.

The LookUp Table

The primary role of the LookUp Table is to provide information such as the current status of connections,the network destination/addresses and throughput information to the *Transport Manager* component. This table is shared with other components such as the *Connection Manager* and the *Streamer*. The structure of the LookUp Table looks as follows:

```
LookUp Table{
        ConnectionID;  // The unique ID assigned to a connection
        Name;          // The session name to which the connection belongs
        Media;         // The media type: either audio or video
        ClientAddress; // The address of the client: IP address or ATM address
        Network;       // The network destination: IP or ATM
        Throughput,     // The throughput taken by that connection
}
```

The LookUp Table is initialized by the *Connection Manager* at connection establishment. It assigns a unique identifier to every media connection, called the ConnectionID.

The Name field is the name of the session. This is an arbitrary text field that the client gives in order to identify its videoconferencing session.

The Media field refers to the type of connection asked by the user; i.e. whether it is an audio or video connection. We assume that data uses the more reliable TCP transport channels and bypasses the stack. This Media field is used by the packet classifier for packet queuing.

The *Transport Manager* uses the ClientAddress field to route the RTP packets towards their respective destinations. This field can be the client IP address or the ATM address with virtual circuit numbers.

The Network field corresponds to the network destination of a media connection i.e. IP or ATM. It is used by the packet classifier for routing RTP packets to their respective queues.

Finally, the throughput field gives an indication of the throughput of every media connection. It is updated on a regular basis by the *Streamer*.

When a connection is closed, the *Connection Manager* resets the fields in the LookUp table. Any pending packets in the queues belonging to a closed connection are discarded by the schedulers.

5.5 Interactions among components

The components of the system either communicate via shared memory or by asynchronous messages. We use shared memory like the LookUp Table for quick updates and where the interaction is simple. Messages are asynchronous to prevent system blocking and unnecessary delays and acknowledgements.

Figure 5.9 illustrates the Message Sequence Chart depicting the connection setup and

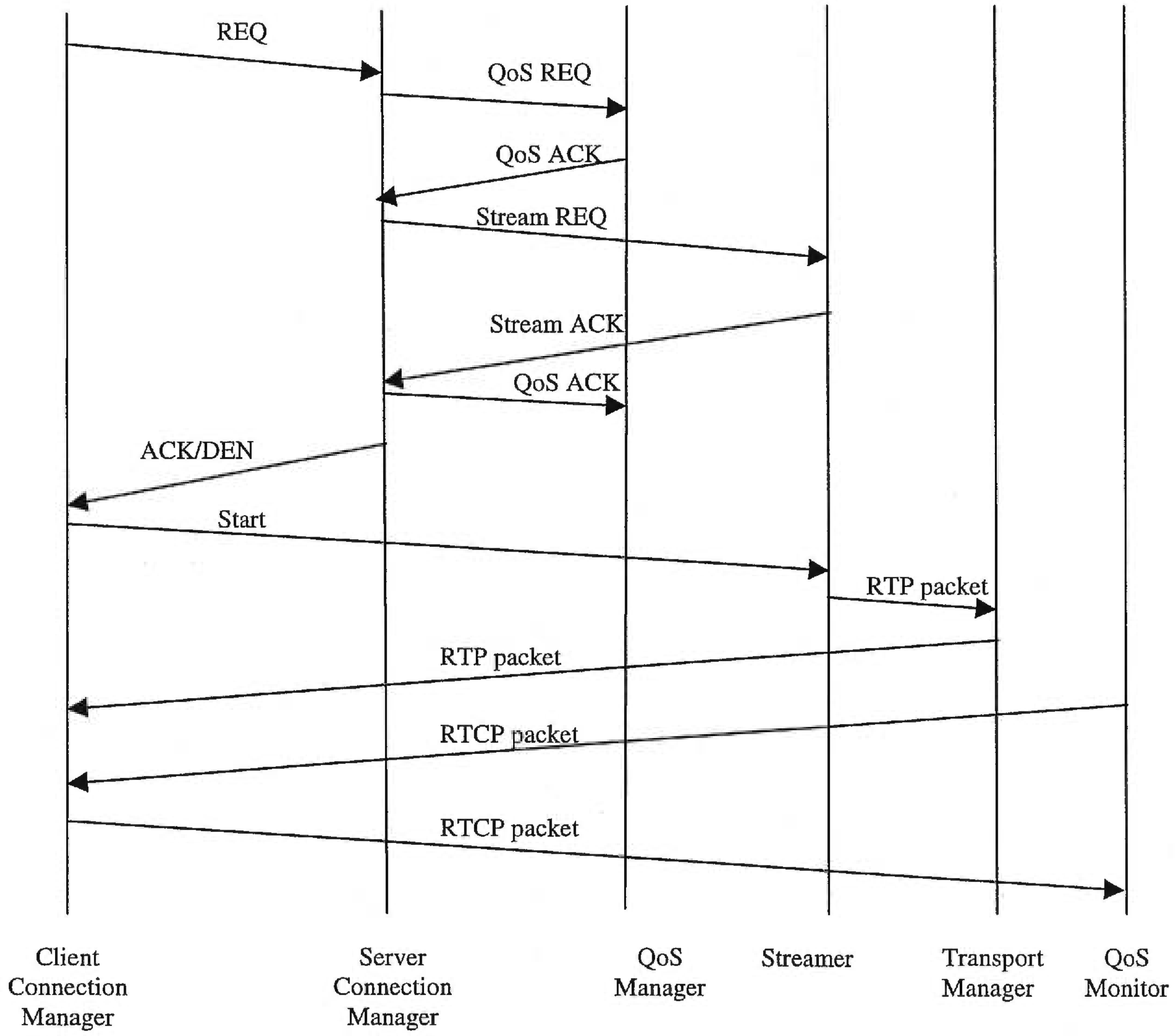


Figure 5.9: Connection establishment and streaming process

streaming process.

The *Connection Manager* on the client side makes a connection request REQ to the server. The request contains the user desired QoS parameters and the group to adhere to.

The *Connection Manager* at the server side accepts requests on a pre-defined port. It relays the request to the *QoS Manager* for admission control. If the user desired QoS levels can be supported, the *QoS Manager* responds with an ACK message to the *Connection Manager*. The latter issues an internal request to the *Streamer* for streaming the required media.

On acknowledgement, the *Connection Manager* informs the *QoS Manager* that the service can be provided. An ACK message with all the necessary information about port numbers at the server side is returned to the client. The latter issues a START message to the *Streamer* to begin streaming the required media.

RTP packets go through the *Transport Manager*, which takes charge of classifying and prioritizing the packets before routing them on the appropriate network to the client.

RTCP packet feedback are exchanged directly between the *QoS Monitor* and the client on pre-defined ports. The feedback is used by the *QoS Manager* to analyze the status of the network connections. It provides the suitable adaptation strategy to deal with any violations in the desired QoS.

This chapter gives a detailed overview of the proposed architecture and its functionality. In the following chapter, we describe its implementation and the results obtained to show the viability of our design choices.

Chapter 6

System Implementation

6.1 Product Interoperability

Most of the videoconferencing applications on the market today are targetted for the PC market. A very small share of applications run under the MacIntosh or UNIX (and its many flavours) operating systems. This can be explained by the growing interest in the Internet and the low connection cost, and decreasing computer prices. We followed the current flow and aimed to implement a tool that runs on top of the Windows operating system, most notably Windows NT 4.0.

To comply by the H.323 standards, we used the Elemedia PX323OS Protocol Stack [url Elemedia] which implements the H.323 family of protocols in a flexible and easy-to-use software package. This protocol stack is a collection of static and dynamic programming libraries that implement the functionality needed to achieve communications within the specifications of the H.323 standard. The facilities provided by the protocol stack are:

- ➢ Remote Access Service (RAS) messages, which are necessary for communications between H.323 endpoints and a gatekeeper.
- ➢ All the mandatory and optional messages, as well as the information elements defined in the call signaling process.

- All call control functions defined by the H.245 standard for in-band signaling. These are used for call establishment, determining capabilities and opening/closing of media channels.
- Support of the RTP/RTCP protocols for media transportation and QoS monitoring.

6.2 Development Platform

The libraries from the Elemedia package use an object-oriented design in C++ and a class-oriented Application Programming Interface (API) is available to simplify application programming complexity. This was conducive to our platform for development since we used Microsoft Visual C++ as programming tool for all the components of the system. We added our own set of APIs for QoS management. Development was made on top of Windows NT 4.0 operating system.

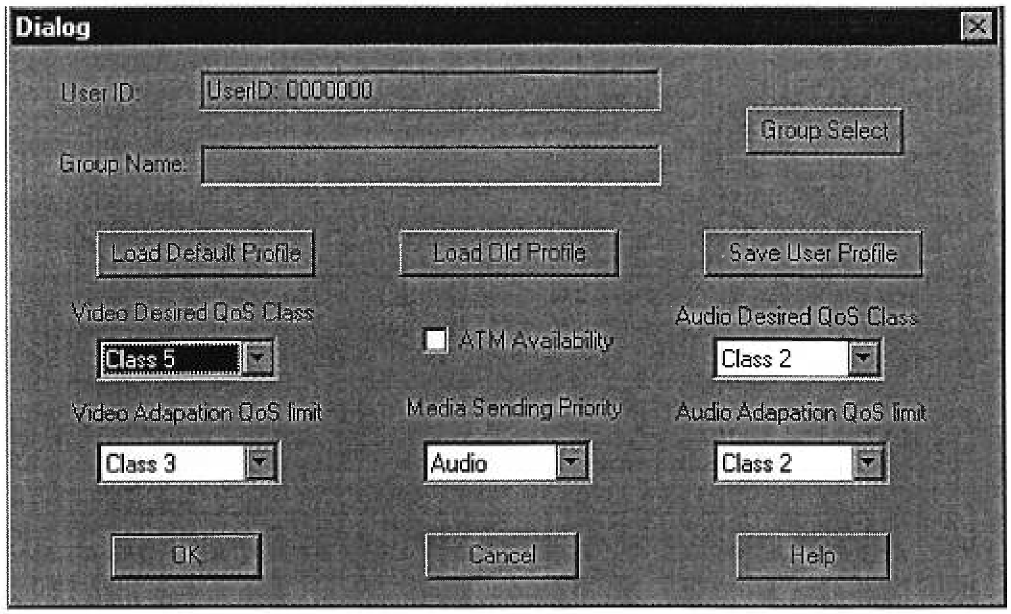
We made an extensive use of Winsock 2.0 socket programming interface. The server module as well as each instance of the client module are encapsulated into a separate Windows process. Each module is based on an multi-threaded architecture and the system components were implemented with threads. This offers an important degree of parallelism and a fine-grain synchronization of multimedia streams.

6.3 Client and Server Interfaces

The interfaces were implemented using the the Microsoft Foundation Class (MFC) Library. MFC is a collection of object-oriented classes written in C++, that provides an overall framework for designing graphical user interfaces for applications, amongst other functionalities.

6.3.1 Client Interface

The main client interface is illustrated in the Figure 6.1.


Dialog

User ID: UserID: 0000000

Group Name:

Group Select

Load Default Profile | Load Old Profile | Save User Profile

Video Desired QoS Class: Class 5

ATM Availability: ☐

Audio Desired QoS Class: Class 2

Video Adapation QoS limit: Class 3

Media Sending Priority: Audio

Audio Adapation QoS limit: Class 2

OK | Cancel | Help

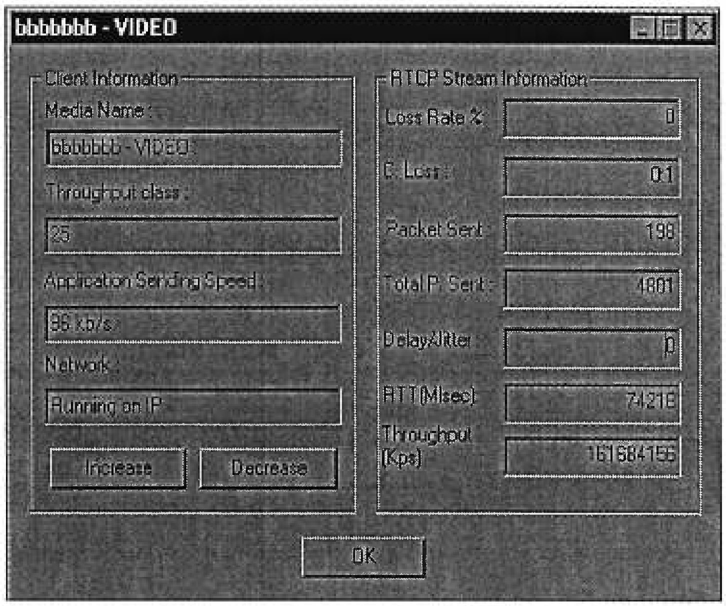
Figure 6.1: Client interface

The client interface allows the user to specify desired and threshold QoS parameters. From the figure, we can distinguish some aspects of this interface namely:

- The user is identified by a self-given name or id. He/She needs to adhere to become a member of a group for the connection to be accepted.
- All the data entered can be saved in a profile on the client side. A saved profile can be loaded in a future use so that the user does not need to enter the same information again.
- The desired and threshold QoS are specified as classes of service, which are indicative of the type of service they expect and tolerate respectively. The mapping of these classes into parameters such as frame rates depend on the codec used which the user can consult on another interface.
- The user specifies the possibility of using the ATM as a secondary network through the check-box "ATM availability".
- The media priority according to the user is specified by the "Media Sending Priority" field on the figure.

6.3.2 Monitoring Interface

The monitoring interface on the server side, provides connection status and QoS parameter information which is provided on a media connection basis. It is illustrated in Figure 6.2.


bbbbbbb - VIDEO

Client Information

Media Name : bbbbbbb - VIDEO

Throughput class : 25

Application Sending Speed : 96 kb/s

Network : Running on IP

RTCP Stream Information

Loss Rate % : 0

C. Loss : 0.1

Packet Sent : 198

Total P. Sent : 4801

Delay/Jitter : 0

RTT(Msec) : 74218

Throughput (Kps) : 161684156

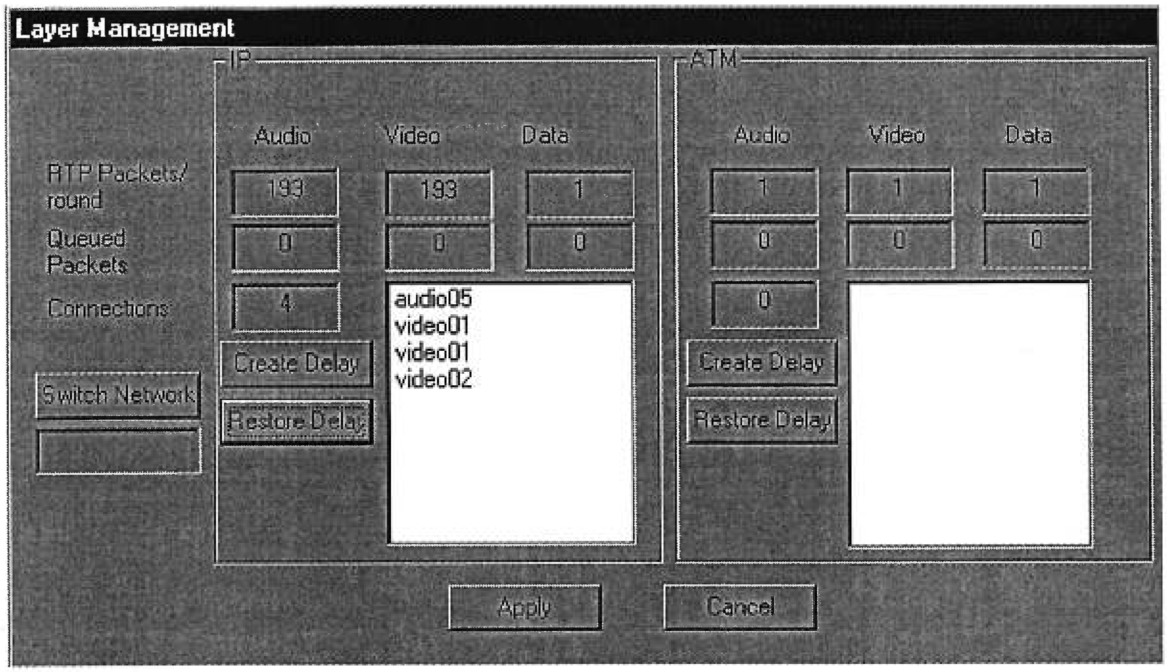
Figure 6.2: Monitoring interface

The left panel displays connection data such as the media name, the throughput class, the speed of transmission from the streamer, and the network on which this connection is established. The Increase and Decrease buttons are used for simulation and will be described in the next section.

The right panel indicates information relating to RTCP packet feedback. QoS parameters such as the loss rate, delay jitter and throughput are displayed. This panel is refreshed every second to give indications of network status in real-time.

6.3.3 The Transport Layer interface

The Transport Layer interface, illustrated in Figure 6.3, shows the management of the multiple network connections by the Transport Manager. There are two panels, with each panel representing RTP packet information for each network. We added the data media type on this figure although conceptually we define data to be encapsulated into the more reliable TCP packets.



	IP Audio	IP Video	IP Data	ATM Audio	ATM Video	ATM Data
RTP Packets/ round	193	193	1	1	1	1
Queued Packets	0	0	0	0	0	0
Connections	4			0		

audio05
video01
video01
video02

Figure 6.3: Transport Layer interface

The following features can be found on the figure:

- The "RTP packets/round" fields indicate the rate at which the scheduler sends packets for each media in one pass onto the network. The rate changes dynamically based on the bandwidth taken by each media connection.
- The "Queued packets" fields give an indication of the amount of RTP packets waiting to be in their respective queues. Their purpose is to serve to analyse packet delays.
- The "Connections" fields gives the number of media connections on the particular network.
- For simulation purposes, the Switch button is used to switch a selected media connection from one network to another. Likewise, the Create and Delay assist in creating bottlenecks and restore the normal flow respectively. Bottlenecks are

created by limiting the RTP Packets/round field and the result can be shown as an increase in the number of queued packets.

6.4 Simulation and Adaptation results

The original plan for our implementation was to have a server and a client machine connected via an IP network and an ATM network. Unfortunately, this configuration did not materalize and we came to the realization that there would be no available ATM switch on which to experiment. We however, have devised generic code for the application to work on a Native-ATM layer. The non-availability of the ATM switch led us to conduct our experiments via simulation on our IP local area network.

The environment consisted of the following: two Pentium Celeron 450 MHZ with 128 MB RAM each acting as server and client machines. The machines are connected on the local Ethernet network and we proceed by simulation to create delays and show the adaptation process.

We present the results obtained from the implementation, particularly the results of the adaptation strategies implemented.

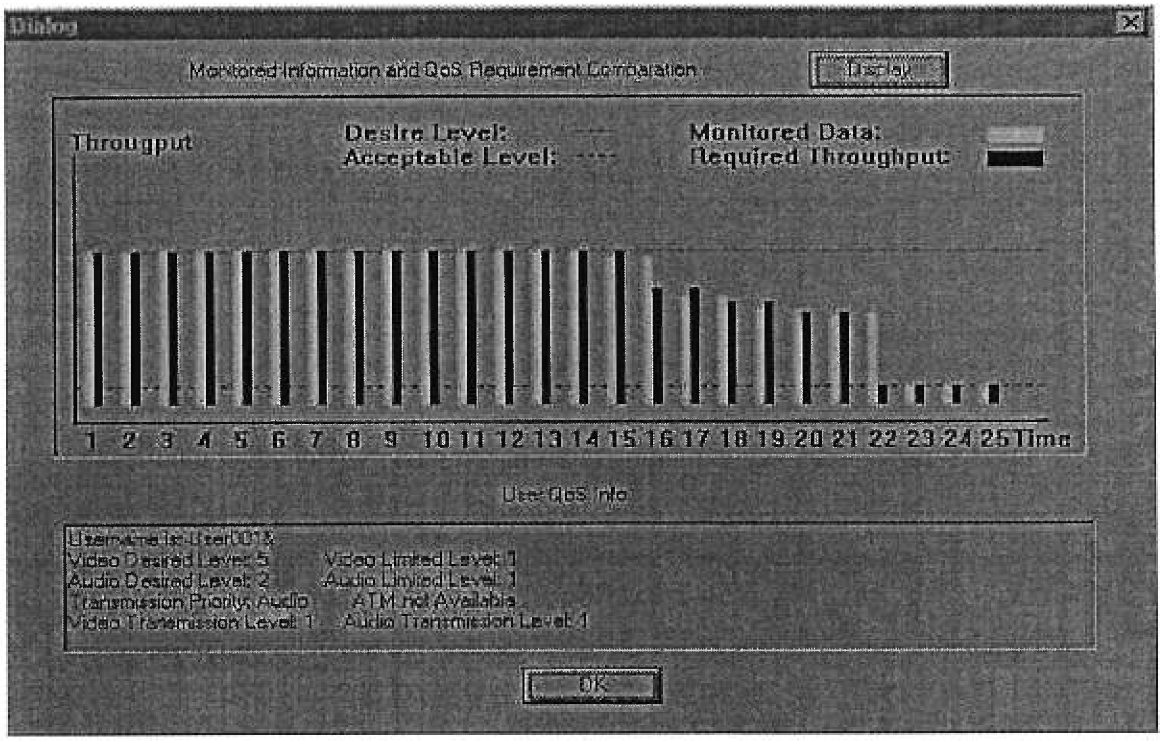
6.4.1 Degradation of QoS

Figure 6.4 shows a simulation of a degradation in QoS resulting from congestion.

The lower panel gives the connection information such as the user session name, the desired QoS levels for audio and video, the currrent QoS provided to the user, the priority of media, and whether the user wants to use the ATM network.

The upper panel portrays a graph of throughput with time. The dark vertical lines represent the required throughput for a QoS level. The light vertical lines represent the actual network throughput at that measured point in time. Time is depicted in terms of RTCP packet feedback; i.e. the graph represents time as a function of the last

25 RTCP packets received. The dotted horizontal lines indicate the desired and tolerable levels of throughput as specified by the client.


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

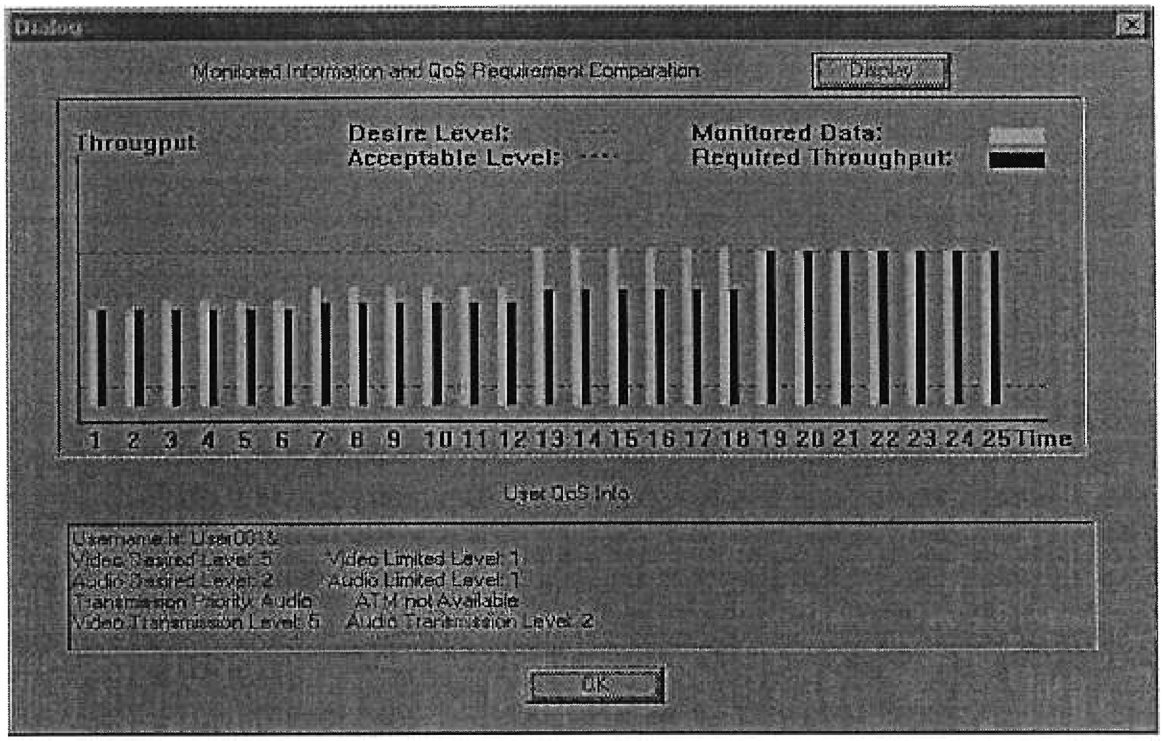
Figure 6.4: QoS Degradation

The following remarks can be extracted from the graph:

- During the first 14 packets, the network can supply the throughput required and the user desired QoS level is satisfied.
- On the 15th feedback packet, the actual monitored throughput is less than the required throughput. The adaptation mechanism is executed.
- On the 16th packet, the QoS level has been decreased and is maintained at that level since the actual throughput is higher than the rquired througput.
- On the 17th packet, congestion continues, resulting in further degradations.
- On the 22nd packet, the threshold QoS level is reached and QoS cannot be decreased any further. Since the user did not select the use of the ATM network as secondary network, the connection is closed.

6.4.2 Upgrading QoS on the same network

The following figure shows the adaptation mechanism increasing the level of QoS to the desired level on the same network, following a period of congestion.


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: User0013
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 6.5: QoS upgrade on the same network

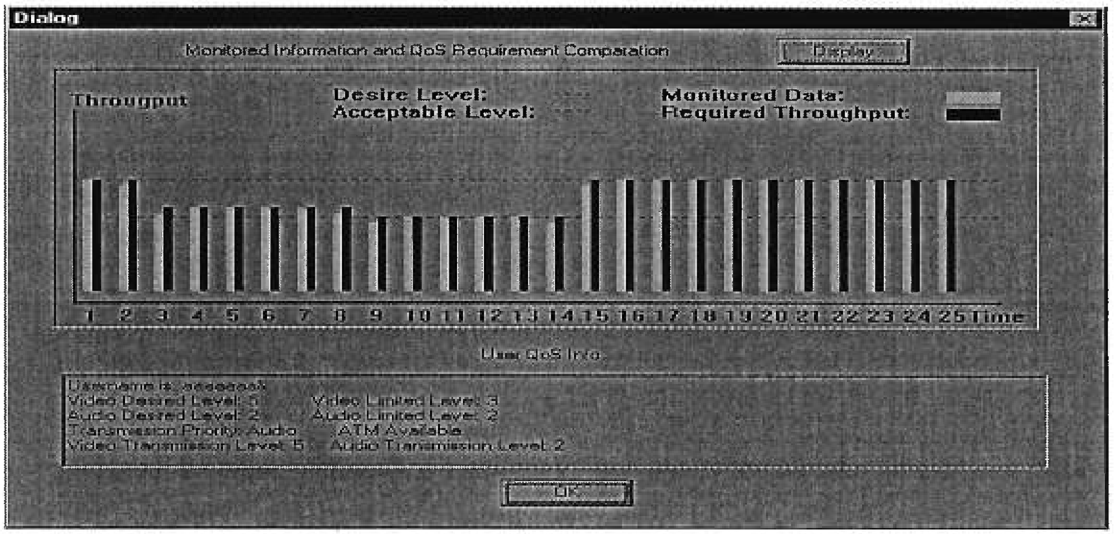
The following remarks can be extracted from the graph:

- On the 3rd packet feedback, the monitored throughput is bigger than the required throughput. The adaptation mechanism does not increase the QoS level right away. In fact, it waits for several reports without violations to increase the QoS level towards the desired level. This is achieved on the 7th packet feedback.
- On the 13th packet feedback, the network can again allow for a higher throughput than the required throughput, therefore the QoS level is increased again.
- Finally as from the 19th packet feedback, the user desired QoS level can be satisfied by the network and the supplied QoS level is increased to that level.

From these two adaptation scenarios, we have shown that increases of QoS level happen at a slower rate than decreases in QoS level. This is conducive to our proposed adaptation algorithm.

6.4.3 Switch from primary to secondary network

In the following figure, we give an example of a network switch when the QoS cannot be maintained on the primary network.


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: [illegible]
Video Desired Level: 5 Video Limited Level: 3
Audio Desired Level: 2 Audio Limited Level: 2
Transmission Priority: Audio ATM Available
Video Transmission Level: 5 Audio Transmission Level: 2

Figure 6.6: Network switch

On the 3nd packet feedback, the QoS offered starts to decrease substantially. The network throughput is very close to the tolerable level.

- On the 9th packet feedback, the network throughput reaches the tolerable level, beyond which point the user desired QoS will not be able to be supported by the primary network.
- On the 14th packet feedback, the network throughput goes beyond the tolerable level. As can be seen from figure 6.6, the user has checked for the possiblity of using the ATM network as secondary network. Hence, all the media connections for that user are switched to the ATM network, with the QoS supplied restored to the user desired level.

- The shift to the ATM network can be seen from the 15th packet feedback. The QoS provided is the user desired level. The QoS stays at that level because the secondary network is not experiencing any congestion.

6.5 Discussion of experimentation and results

The results obtained on each time measurement, as represented by the bar graphs, were as expected. They show the correctness of the implemented adaptation algorithms in terms of QoS degradation, upgrade and network switching as well as the viability of the proposed concepts. The following further remarks can be pointed out from the experimentation namely:

- To our knowledge, our work is the first QoS architecture to date that implements the notion of network switch as part of an adaptation strategy. Therefore it is difficult to make comparisons with other architectures on that basis. We note that comparisons with other architectures in terms of response times, delays, packet losses, picture quality as depicted by the client etc., are possible if the source codes for the latter are freely available for implementation. The non-heterogeneity of the platforms used and non standardization of other architectures add to the complexity of achieving such comparisons.
- The simulation results presented have been obtained from performance tests conducted at different periods of time in local area network with star topology running on 100 Mbps fiber optic links, with a central hub which generally works at 75% of its capacity. Although this test environment does not exactly reflect the traffic conditions on the Internet, the heterogeneity of the computers and applications used as wells as the traffic induced by our tool provide an approximation.
- The current implementation does not support any real multimedia streams such as real video frames and audio. We were mainly constrained by the exorbitant prices of H.323 codecs available on the market. We thus had to test the application with raw generated data.

- While our architecture lays the groundwork for on-line teaching, we acknowledge that much work remains to be done to achieve a full implementation of tele-teaching. We have started with the implementation of workgroups but other functionalities such as floor control, group synchronization, information stocking etc. need to be implemented.
- Understanding the functioning of the Elemedia package implementing the H.323 stack turned out to be a quite challenging task. The lack of proper documentation and the complexity of implementing the H.225 and H.245 protocols led us to create lighter-weight protocols for connection establishment. This entails issues about H.323 compliance.

In the following chapter, we provide concluding remarks by giving a summary of the thesis and we address a number of concerns and enhancements that can be applied to the proposed architecture as part of ongoing and future work.

Chapter 7

Conclusion

7.1 Thesis Summary

For the past years, a wide range of multimedia applications have become increasingly available to the personal computing environment. A number of factors have contributed to this evolution namely: the advent of the Internet, faster and cheaper processors, and drastic changes in the telecommunications industry among others.

Examples of such applications are videoconferencing systems, distance or on-line training, multimedia information systems, geographical information systems, video-on-demand etc. The main characteristics of such systems is the incorporation of continuous media such as video, voice and animation. They require continuous data transfer over a certain lapse of time, such as the play-out of a video stream from a remote camera to a group of users across a computer network.

One of the principle challenges facing these multimedia applications is to provide some guarantees so as to satisfy the requirements of the user. The current Internet, being a "best effort" service provider, cannot give such guarantees. It processes traffic as quickly as possible but cannot guarantee timeliness and delivery. Researchers have been and are still working on techniques to provide that degree of service. The term Quality of Service (QoS) is the keyword commonly used to represent the set of requirements imposed by a user related to the performance to be achieved in the provision of services by an application.

The management of QoS is essential in providing QoS. It is mainly a set of activities that work towards finding appropriate QoS characteristics of a multimedia system, to achieve its required functionality and optimize system performance. Much work has been done in the context of standardized frameworks to provide QoS within distributed systems and communication networks.

We have presented three models proposed by the Internet Engineering Task Force (IETF), namely the Integrated Services model, the Differentiated Services model, and the Multi-protocol Label Switching model, that provide mechanisms for meeting the demand for QoS at the core of the Internet.

We have also described three QoS architectures that formalize QoS in end-systems and networks by providing a framework that integrates QoS control and management mechanisms. These architectures are namely the OMEGA architecture, the Extended Reference Model (XRM), and the QoS-A model.

Most of research related to guaranteeing QoS has been focused on resource reservation in communication networks such as ATM. The applicability of reserving resources for the Internet is a topic of much debate because its present architecture is founded on the fact that all flow-related state information is stored in end-systems. One complementary solution to resource reservation is to design adaptive applications that become more tolerant to inevitable fluctuations from the network. We have presented several mechanisms of how adaptation mechanisms can be implemented in multimedia applications in Chapter 4.

For the past couple of years, we have witnessed a growing consensus from industry professionals, including vendors, for the need to build multimedia applications based on industry standards. A number of standards have been ratified by prominent organizations such as ITU, to meet interoperability and compatibility concerns. The H.323 standard is the industry standard for providing a foundation for audio, video and data communications across networks such as the Internet, that cannot provide QoS guarantees. We have given a detailed description of the H.323 standard and the components and protocols it specifies.

We have addressed all the challenges mentioned by proposing an H.323-compliant architecture that provides a framework for QoS management and adaptable to QoS fluctuations. Our design was based on an on-line training case study. Our architecture allows the specification of user requirements, performs the formalization and mapping of these requirements into system and network parameters. Our proposed adaptation strategies react to network feedback obtained from the Real Time Transport Protocol (RTP), which is supported by the H.323 specification. We have introduced the concept of multiple network connections in which a client and a server are connected via more than one network. The choice of multiple networks has an important impact on our adaptation choices. A transport layer has been defined to manage these multiple network connections, and to work independently of the underlying network protocol. The overall architecture and design choices have been described in Chapter 5.

In Chapter 6, we have presented the implementation of a prototype tool for our architecture. We used the Elemedia H.323 stack for H.323 communications and RTCP feedback. The interfaces and simulation results have been illustrated, showing the working of the various system components. The execution of the adaptation strategies implemented were shown by the means of graphs, and the results were as expected, showing the viability of our strategies and design choices.

The implemented architecture was conducted by a team of three people. On our part, we focused mainly on the adaptation strategies and the mechanisms behind the Transport Manager as well as the integration of all system components.

7.2 Future Work

A number of considerations and enhancements can be made to our work to overcome its limitations and develop it into a fully functioning product. We present them below, with no order of preference or importance.

The synchronization of media needs to be addressed for reconstructing media streams at the client side. There are widely-used models about media synchronization that have been proposed in the literature [Owezarski *et al.* 95]. Its need becomes even more important in a multiple network configuration where media can be coming from different sources.

Further enhancements can be conducted on the management and control of session groups in the cooperative work. Areas such as security mechanisms for guaranteeing data confidentiality pertaining to particular groups, multi-call setups, dynamic allocation of resources and administration of rights are some features that can be added to our prototype tool [Hardman *et al.* 98].

The choice of the Windows operating system as development platform for our prototype tool was mainly a marketing issue. The overwhelming accessibility to PC's running the Windows family of operating systems led us to that choice. The base operating system however does not provide native support for real-time applications. However, there is the issue of cost to consider while using real-time operating systems.

The concept of IP multicast [Mittra 97] to take advantage of the MBONE and multicast servers is another area to consider. Multicast provides more efficient routing capabilities and bandwidth consumption, particularly when there are many participants in a teleconference. With the advent of IPv6 and its support for multicast as a native communication mode [Braun 97], more and more multimedia applications will provide multicast facilities.

A lot of debate nowadays concerns the pertinence of the H.323 standard to provide the interoperability sought for. A growing amount of interest seems to be geared towards the SIP protocol which represents a different approach to the same problem that H.323 addresses. Proponents of SIP argue that the latter is a lightweight protocol that is relatively easy to implement compared to the complexity of H.323, provides a rich extensibility and better scalability compared to H.323. However, further work needs to be conducted to evaluate the differences in quantitative performance of these two protocols.

Finally, QoS routing or constraint-based routing techniques [Wang & Crowcroft 96] can be seen as a generalization of our strategies for network selection. While computing network paths, these techniques not only consider network topology, but also requirements of flows, resource availability of links and other policies specified by network administrators. The addition of information on link states to compute routes will add to the complexity of our algorithms, but might provide more efficiency in network utilization.

Bibliography

[Alfano 97]	Alfano M. (1997). User Requirements and Resource Control for Cooperative Multimedia Applications. *ECMAST 97*. Fdida S. & Morganti M., eds., Springer-Verlag. 1997. pp 537-552.
[Blake *et al.* 98]	Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W. (1998). An Architecture for Differentiated Services. *RFC 2475*. December 1998.
[Bochmann & Hafid 96]	Bochmann G. & Hafid A. (1996). Some Principles for QoS Management. *The International IFIP Workshop on QoS*. Paris, France. 1996.
[Bochmann *et al.* 96]	Bochmann G., Kerhervé B., Hafid A., Dini P. & Pons A. (1996). Architectural Design of Adaptive Distributed Multimedia Systems. *Proceedings of the IEEE International Workshop in Distributed Multimedia Systems Design*. Berlin, Germany. 1996.
[Bolot *et al.* 94]	Bolot J. C., Turletti T. & Wakeman I. (1994). Scalable Feedback Control for Multicast Video Distribution in the Internet. *Proceedings of ACM SIGCOMM 94*, **24**. October 1994. pp 58-67.
[Braden *et al.* 94]	Braden R., Clark D., Shenker S. (1994). Integrated Services in the Internet Architecture: an Overview. *RFC 1633*. June 1994.
[Braden *et al.* 97]	Braden R., Zhang L., Berson S., Herzog S., Jamin S. (1997). Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification. *RFC 2205*. September 1997.

[Braun 97]	Braun T. (1997). Internet Protocols for Multimedia Communications. *IEEE Multimedia*, **4**. No. 3. July-September 1997. pp 85-90.
[Busse *et al.* 95]	Busse I., Deffner B. & Schulzrinne H. (1995). Dynamic QoS Control of Multimedia Applications based on RTP. *First International Workshop on High Speed Networks and Open Distributed Platforms.* St. Petersburg, Russia. June 1995.
[Campbell & Coulson 96]	Campbell A. & Coulson G. (1996). A QoS Adaptive Transport System: Design, Implementation and Experience. *ACM Multimedia.* Boston, MA. November 1996. pp 117-127.
[Campbell *et al.* 93]	Campbell A., Coulson G. & Hutchison D. (1993). A Multimedia Enhance Transport Service in a Quality of Service Architecture. *Workshop on Network and Operating System Support for Digital Audio and Video.* Lancaster, UK. 1993. pp 124-137.
[Campbell *et al.* 94]	Campbell A., Coulson G. & Hutchison D. (1994). A Quality of Service Architecture. *ACM Computer Communications Review*, **24**. No. 2. 1994. pp 6-27.
[Cho & Shin 99]	Cho S. & Shin Y. (1999). Multimedia Service Interworking over Heterogeneous Networking Environments. *IEEE Network,* **13**. No. 2. March/April 1999. pp 61-69.
[Cruz 91]	Cruz R. (1991). A Calculus for Network Delay, part II: Network Analysis. *IEEE Transactions on Information Theory*, **37**. No. 1. March 1991. pp 132-141.
[Dan *et al.* 95]	Dan A., Kienzle M. & Sitaram D. (1995). A Dynamic Policy of Segment Replication for Load Balancing in Video-on-Demand Servers. *ACM Multimedia Systems Journal*, **3**. 1995.
[El-Marakby & Hutchison 96]	El-Marakby R. & Hutchison D. (1996). Evaluation of the Real-Time Transport Protocol (RTP) for Continuous Media Communications. *Proceedings of the 3rd Communication Networks Symposium.* Manchester, UK. July 1996. pp 188-191.

[El-Marakby & Hutchison 97]	El-Marakby R. & Hutchison D. (1997). Delivery of Real-time Continuous Media over the Internet. *Proceedings of the 2nd IEEE Symposium on Computers and Communications*. Cairo, Egypt. 1997.
[Eriksson 94]	Eriksson H. (1994). MBONE: The multicast backbone. *Communications of the ACM*, **37**. August 1994. pp 54-60.
[Ferrari *et al.* 92]	Ferrari D., Banerjea A. & Zhang H. (1992). Network Support for Multimedia. *International Computer Science Institute Technical Report 92-072*. November 1992.
[Gecsei 97]	Gecsei J. (1997). Adaptation in distributed Multimedia Systems. *IEEE Multimedia,* **4**. No. 2. April-June 1997. pp 58-66.
[Hafid & Bochmann 97a]	Hafid A. & Bochmann G. (1997). An Approach to QoS Management in Distributed MM Applications: Design and an Implementation. *Multimedia Tools and Applications Journal*. 1997.
[Hafid & Bochmann 97b]	Hafid A. & Bochmann G. (1997). Quality of Service Adaptation in Distributed Multimedia Applications. *ACM Multimedia Systems Journal*. 1997.
[Hafid *et al.* 97]	Hafid A., Bochmann G. & Dssouli R. (1997). Quality of Service Negotiation with Present and Future Reservations (NAFUR). *Computer Networks and ISDN Systems Journal*. 1997. pp 777-794.
[Hardman *et al.* 98]	Hardman V., Sasse M. A. & Kovelas I. (1998). Successful multiparty audio communication over the Internet. *Communications of the ACM*, **41**. No. 5. May 1998. pp 74-80.
[Huard *et al.* 96]	Huard J. F., Inoue I., Lazar A. A., Yamanaka H. (1996). Meeting QoS Guarantees by End-to-End QoS Monitoring and Adaptation. *Proceedings of Workshop on Multimedia and Collaborative Environments of the 5th IEEE International Symposium On High Performance Distributed*

Computing. Syracuse, USA. August 1996.

[Hutchison *et al.* 94] Hutchison D., Coulson G., Campbell A. & Blair G. (1994). QoS Management in Distributed Systems. *Network and Distributed Systems Management*. Morris Sloman, ed., Addison-Wesley. 1994. pp 273-303.

[ITU-T H.245 96] H.245: Control Protocol for Multimedia Communication. *ITU-T Recommendation*. March 1996.

[ITU-T H.323 98] H.323: Packet-Based Multimedia Communications Systems. *ITU-T Recommendation*. February 1998.

[ITU-T T.120 96] T.120: Data protocols for Multimedia Conferencing. *ITU-T Recommendation*. July 1996.

[Kanakia *et al.* 93] Kanakia H., Mishra P. & Reibman A. (1993). An adaptive congestion control scheme for real-time packet video transport. *SIGCOMM Symposium on Communications Architectures and Protocols*. San Francisco, CA. Sept 1993. pp 20-31.

[Kostas *et al.* 98] Kostas T. J., Borella M. S., Sidhu I., Schuster G. M., Grabiec J. & Mahler J. (1998). Real-Time Voice Over packet-Switched Networks. *IEEE Network*, **12**. No. 1. January/February 1998. pp 18-27.

[Kurose & Ross 99] Kurose J. & Ross K. (1999), Computer Networking and Internet Protocols. Addison-Wesley. 1999.

[Lazar *et al.* 96] Lazar A. A., Lim K. & Marcocini F. (1996). Realizing a Foundation for Programmability of ATM Networks with the Binding Architecture. *IEEE Journal of Selected Areas in Communication*, **7**. September 1996. pp 1214-1227.

[McCanne & Jacobson 95] McCanne S. & Jacobson V. (1995). vic: A Flexible Framework for Packet Video. *ACM Multimedia*. San Francisco, CA. November 1995.

[Mills 92] Mills D. (1992). Network Time Protocol (v3). *RFC 1305*. April 1992.

[Mittra 1997]	Mittra S. (1997). Iolus: A framework for scalable secure multicasting. *ACM Computer Communication Review*, **27**. October 1997. pp 277-288.
[Moran & Wolfinger 92]	Moran M. & Wolfinger B. (1992). Design of Continuous Media Data Transport Service and Protocol. *University of California Technical Report TR-92-019*. 1992.
[Nahrstedt & Smith 95a]	Nahrstedt K. & Smith J. M. (1995). Design, Implementation and Experiences of the OMEGA Architecture. *University of Pennsylvania Internal Report MS-CIS-95-22*. May 1995.
[Nahrstedt & Smith 95b]	Nahrstedt K. & Smith J. M. (1995). The QoS Broker. *IEEE Multimedia*, **2**. No. 1. Spring 1995. pp 53-67.
[Nahrstedt 95]	Nahrstedt K. (1995). An Architecture for End-to-End Quality of Service Provision and its Experimental Validation. *University of Pennsylvania Ph.D. Thesis*. 1995.
[Nichols *et al.* 98]	Nichols K., Blake S., Baker F., Black D. (1998). Definition of the Differentiated Services Field (DS Field) in the IPv4 and Ipv6 Headers. *RFC 2474*. December 1998.
[Nilsson & Karlsson 97]	Nilsson S. & Karlsson G. (1997). Fast Address Lookup for Internet Routers. *Proceedings of ACM SIGCOMM 97*. Cannes, France. September 1997.
[OMG CORBA 96]	The Common Object Request Broker: Architecture and Specification. *OMG Document ptc/96-03-04*. 1996.
[Owezarski *et al.* 95]	Owezarski P., Diaz M. & Sénac P. (1995). Modélisation et implémentation de mécanismes de synchronization multimédia dans une application de visioconférence. *Colloque francophone sur l'ingéniérie des protocoles*. Rennes, France. mai 1995.

[Parris *et al.* 93] Parris C., Ventre G. & Zhang (1993). Graceful Adaptation of Guaranteed Performance Service Connections. *International Computer Science Institute Technical Report TR-93-011*. Berkeley, USA. November 93.

[Ramanathan & Rangan 93] Ramanathan S. & Rangan P. V. (1993). Adaptive feedback techniques for synchronized media retrieval over integrated networks. *IEEE/ACM Transactions and Networking*, **1**. No. 1. January 1993.

[Ravindram & Steinmetz 93] Ravindram K. & Steinmetz R. (1993). Transport-Level Abstractions for Multimedia Communications. *IEEE INFOCOM 93*. San Francisco, CA. March 1993.

[Reinema *et al.* 98] Reinema R., Tietze D. A. & Steinmetz R. (1998). IMCE – An Integrated Multimedia Conferencing and Collaboration Environment. *Proceedings of the 1st National CSCW Workshop*. Beijing, China. December 1998. pp 95-100.

[Robert *et al.* 97] Robert L., Gaussens D. & Villemur T. (1997). Etat de l'art sur le Télé-Enseignement. *Rapport LAAS No. 97226.* juin 1997.

[Romanow & Floyd 95] Romanow A. & Floyd S. (1995). Daynamics of TCP traffic over ATM networks. *IEEE JSAC*, **13**. No. 4. May 1995. pp 633-641.

[Rosen *et al.* 98] Rosen E., Viswanathan A. & Callon R. (1998). Multiprotocol Label Switching Architecture. *Internet draft, draft-ietf-mpls-arch-01.txt*. March 1998.

[Rosenberg & Schulzrinne 98] Rosenberg J. & Schulzrinne H. (1998). Timer reconsideration for enhanced RTP scalability. *Proceedings of the Conference on Computer Communications*. San Francisco, CA. April 1998.

[Schulzrinne & Rosenberg 99] Schulzrinne H. & Rosenberg J. (1999). Internet telephony: Architecture and protocols – an IETF perspective. *Computer Networks and ISDN Systems*, **31**. February 1999. pp 237-255.

[Schulzrinne 95]	Schulzrinne H. (1995). Internet Services: from Electronic Mail to Real-Time Multimedia. *Kommunikation in verteilten Systemen*. Chemnitz, Germany. February 1995. pp 21-34.
[Schulzrinne *et al.* 90]	Schulzrinne H., Kurose J. F. & Towsley D. (1990). Congestion Control for Real-Time Traffic in High-Speed Networks. *IEEE INFOCOM 90*, **2**. San Francisco, CA. Jun. 1990. pp 543-550.
[Schulzrinne *et al.* 94]	Schulzrinne H., Kurose J. & Towsley D. (1994). An evaluation of scheduling mechanisms for providing best-effort, real-time communication in wide-area networks. *IEEE INFOCOM 94*. Toronto, CA. June 1994.
[Schulzrinne *et al.* 96]	Schulzrinne H., Casner S., Frederick R. & Jacobson V. (1999). RTP: A Transport Protocol for Real-Time Applications. *RFC 1889.* January 1996.
[Schulzrinne *et al.* 98]	Schulzrinne H., Rao A., Lanphier R. (1998). Real time streaming protocol (RTSP). *RFC 2326*. April 1998.
[Semeria 00]	Semeria C. (2000). Multiprotocol Label Switching: Enhancing Routing in the New Public Network. *Juniper Networks White Paper*. March 2000.
[Shenker *et al.* 97]	Shenker S., Partridge C. & Guerin R. (1997). Specification of Guaranteed Quality of Service. *RFC 2212*. September 1997.
[Tehrani 00]	Tehrani R. (2000). Interoperability, Taking Wing, And a Prayer. *Internet Telephony*, **3**. No. 1. January 2000. pp 8-10.
[Toga & ElGebaly 98]	Toga J. & ElGebaly H. (1998). Demystifying Multimedia Conferencing Over the Internet Using the H.323 Set of Standards. *Intel Technology Journal*. Q2. 1998.
[url ATM]	http://www.atmforum.com
[url Elemedia]	http://www.elemedia.com

[url IETF]	http://www.ietf.org
[url ITU-T]	http://www.itu.ch
[url MPEG]	http://www.mpeg.org
[url NetMeeting]	http://www.netmeeting.com
[url VDP]	http://choices.cs.uiuc.edu/Papers/New/vosaic/vosaic.html
[Verma *et al.* 91]	Verma D., Zhang H. & Ferrari D. (1991). Delay Jitter Control for Real-Time Communication in a packet switching network. *Proceedings of IEEE TRICOMM.* April 1991.
[Verma *et al.* 91]	Verma D., Zhang H. & Ferrari D. (1991). Delay Jitter Control for Real-Time Communication in a packet switching network. *IEEE TRICOMM 91.* 1991.
[Vogel *et al.* 95]	Vogel A., Kehervé B., Bochmann G. & Gecsei J. (1995). Distributed Multimedia Applications and Quality of Service: A survey. *IEEE Multimedia,* **2**. No. 2. Summer 1995. pp 10-19.
[Wakeman 93]	Wakeman I. (1993). Packetized video – Options for interaction between the user, the network and the codec. *The Computer Journal*, **36**. No. 1. 1993. pp 55-67.
[Wang & Crowcroft 96]	Wang Z. & Crowcroft J. (1996). Quality of Service Routing for Supporting Multimedia Applications. *IEEE JSAC*. September 1996.
[Wee *et al.* 96]	Wee S. J., Polley M. O. & Schreiber W. F. (1996). A Generalized Framework for Scalable Video Coding. *Multimedia Communication and Video Coding*. Wang Y. et al., eds., Plenum Press. New York, USA. 1996. pp 483-490.
[Wroclawski 97]	Wroclawski J. (1997). Specification of the Controlled-Load Network Element Service. *RFC 2211*. September 1997.
[Xiao & Ni 99]	Xiao X. & Ni L. M. (1999). Internet QoS: A Big Picture. *IEEE Network,* **13**. No. 2. March/April

1999. pp 8-19.

[Yin & Hluchyj 91] Yin N. & Hluchyj M. G. (1991). A Dynamic Rate Control Mechanism for Integrated Networks. *IEEE INFOCOM 91*, **2**. 1991. pp 543-552.

[Zhang & Ferrari 94] Zhang H. & Ferrari D. (1994). Rate-Controlled Service Disciplines. *Journal of High Speed Networks*, **3**. No. 4. 1994.

[Zhao *et al.* 00] Zhao W., Olshefski D. & Schulzrinne H. (2000). Internet Quality of Service: an Overview. *Columbia University Draft Paper No: CUCS-003-00*. New York, USA. February 2000.