A New Paradigm for Audio Conferencing on Voice over IP (VoIP)

A Thesis
Submitted for the Degree of
Doctor of Philosophy
in the Faculty of Engineering

By
R. Venkatesha Prasad

Centre for Electronics Design and Technology
Indian Institute of Science
Bangalore – 560 012, India

July 2003

Confession

I know this section is not meant for evaluation. So! Why not I write the truth. Yes! Because I stand for Truth. I write TRUTH and nothing but *Truth* on Oath *here*[1].

I want to absolve myself of the criminal proceedings that may lead to rigorous punishment for submitting and publishing this piece of *Work* ("Work" means *this thesis* hereinafter). This work should be buried deep inside the ground because it bamboozles readers and may make them slaves of *the* technology. It may discourage junta travelling to distant places to see friends, make friends, interact with peers and enjoy the Conferences and Meetings! It is harmful for Tourism, Airlines and many hotels who serve customers like Gods! It is a social crime to nobble (not noble!) the livelihood of so many people. To the Government, which gave me public funds for so many years to carry out research I am giving back this piece of *Work*. I am giving back something - if it is made public - that will lessen the income, which would have been otherwise thrived upon through international Tourism!

I am submitting this "tender of pardon to accomplice" because I hope that I may be acquitted (awarded). Section 306 of CrPC reads as under:

> "... (the person who is) trying the offence, at any stage of the inquiry or trial, may tender a pardon to such person on condition of his making a full and true disclosure of the whole of the circumstances within his knowledge relative to the offence and to every other person concerned, whether as principal or abettor, in the commission thereof."

At first sight, though it looks like I am the principal (or even the sole) accused party of the crime, I plead guilty of being *only* a means. I implore you to take me as an approver for the prosecution during trial (defence) of the crime which is being disclosed here.

The first person I want to name, who is responsible more than me is Prof. H S Jamadagni, conspirator (my research supervisor). It is not appropriate if I just say that he guided me throughout. He thinks pretty differently than many others in the field and with both feet on the ground. His way of research is to pick a problem, which is hot and useful. He often remarks "... it's important to work on what is to be solved than what can be solved". The ideas, field of work, and technological and product oriented look at the problems are a few cases in point. If I look back at the path I covered in these years, there are very few occasions in which I followed him. In most of the circumstances, when I was languishing, he carried me all along. He has *a hand* in everything that is being reported here.

The next, not less by any chance, is Dr. Joy Kuri my unofficial co-guide. He is one of the pillars (there were only two, the other one you know by now!) of the project I worked on during and/or for my thesis work. The discussions I had with him were nothing but crucial. If you see some significant points expressed here it is because of him. He motivated me to do many things that I had neglected (it's natural for me!) during the course of this work. Many a time he took out three different (coloured) pens (pen is mightier than sword!!) and

[1] I didn't say the remaining pages don't contain truth!

jotted down (just short of Tex-ing it) for me. He too has *a hand* in all the contribution of this report. It is said that *two* hands are more than enough to do any celestial act. I have told you about the *hands*[2] behind the scenes, now, it is simple to collect the evidences. Just see the report in your hands it is easy to find out their crafty works. Now tell me do you want me to exclude them?

It's not over; Prof. Y Narahari of CSA department is instrumental in making me pursue higher studies (this offence! without *stopping*) and guided me onto this path before handing me over to the above two. He very well knew then what was going to happen. I would have dropped his name if it were to be only this much. He constantly kept in touch and heard patiently all that I blabbered. He picked up, from time to time, the real grains for me. His ideas are there to see. There are two other persons who pushed me into this, Prof. Utpal Mukherjee and Anandji of ERNET, IISc. I don't intend to say anything least about Prof. Anurag Kumar who gave me a chance to work in the field of Networks though it was not my field of expertise then[3] when I was awfully down with depression. I wouldn't have even tried this work without their wholehearted support.

You see for a well-planned show you need dress rehearsals. All my ideas are first disclosed before a *sounding board* to find the flaws in the plotting and that is Sri. T V Prabhakar. You may say that he is passive. He did more than this. He proactively arranged the scattered data I used to throw at him and kept on accumulating it for me. It is not only this much, you ask him any thing, you get it in 30min or less. All the facilities and help I had during the commission of this crime are no less than my execution of the same. Do you want to exclude this co-conspirator?

This work is related to a sponsored[4] project. Sri. Ravi A Ravindranath and Prof. Kumar N Shivarajan were very actively involved in this work from the beginning. This offence was committed in many smaller stages; Ravi is responsible for propelling me to the orbit - where I am now - with many stages of boosters. Messrs. Ramakrishna Potti, Niranjan, Deepak and Hari Bhaskaran have all been no meaner in this regard.

Messrs. Haresh Dagale, *the Witty*, Varchas *the Bellator*, Prasanna *the Mimicker*, Anand *the Dwaitin*, and Rajesh *the Worker* were my project mates who have contributed immensely. This five men army was in the forefront shielding me. Alas! if they had not done that I would never have been writing this piece indicting them. They are reaping the fruit of their labour! They weigh up equally with others I have mentioned. New entrants are Samar and Abid. I would like to leave them since I know they are going to take the same path (comraderie!). Richard Hurni *the Professional* is the latest one but he is not in India to bring him to the box! The prosecution has to apply for deportation from Switzerland. Finally, my project students Abhijeet, Chiranth, Rahul and Vishal, who are responsible for many measurements and implementations (executions!) I did.

Now the Abettors:

My interaction with Dr. Andre Pittet was one of the nice things that could have happened to any budding research student (who went on to become an offender). You can see his handy work in the thesis that has some real (*felonious*?) thought provoking quotes.

I remember all my interactions with Padmashri Prof. N. Balakrishnan of SERC and the chairman of Information Sciences division. The computing facilities and some discussions with him in the initial stages of my existence here in the campus was a shot in the arm. You go and ask him for a *root* account at 1AM past midnight and lo! you get it the next day. I

[2]Similar to Maradona's famous "hand of God !" in '86 soccer semis against England.

[3]It is not even now, if you think there is some contribution in this work, you know who the real contributors are! and you know whom to catch.

[4]See *Godfather* by Mario Puzo.

can feel his helping hand in all the data I wanted to collect and the disk space for committing this act. As one of my friends put it, he is *simple* and *efficient* and that is Balki. Nay! balki! He pampered me a lot in all my non-academic endeavours. If you want to avoid others to take my path, you know who is the root cause!

Prof. Vittal Rao of Maths, Prof. K J Rao of SSCU, Prof. H S Mukunda of Aero Space, and Prof. K B R Verma of MRC have all kept me alive with many interdisciplinary work. Including my guide, they have all tolerated my indiscipline(ary) work under the cover of interdisciplinary work. Prof. Vittal Rao taught me analysis and if you see any (un)structured arguments in this work, you know why! Prof. K J Rao in particular, who was the prime mover, kept my confidence level on par with some of the greatest leaders (criminals) ever. I wouldn't have done it and even this confession so boldly without their constant encouragement.

CEDT faculty and staff; Prof. Sinha, Prof. Gopakumar, Prof. N J Rao, Dr. Gunashekaran, Dr. Umanand, Kruvilla Vergheese, MKK, Dr. Dinesh, NVCR, Muthu Kumar, G V Mahesh, Nagarathna, Kalim, Saravanan, Kalpana, Singhji, Gururaj, Shivaswamy, and Gopalakrishna have all nurtured this child who has become a juvenile hardcore now. You give them a raw innocent fellow and you know by my example how they can transform him. You want to drop them out of this list? Nay! Not even one bit, they are all responsible for me to be in this situation. Some of them selected me and some of them fed me!

Smt. Laxmi Raj of office arranged money whenever I needed. In a sense she is the means for funding all my (underground?) activities (un)knowingly. If you want some financial support just ask her. I want to exclude her name (you see, I need some funds after I get acquitted!).

Smt. Prabha Patil, the meticulous librarian who gave me books whenever I asked, thinking that I'd have read them (Please don't break her heart by telling her that I had no time to spend on all those good(?!) activities!). I want to name the supporting cast Smt. Parvathi too. Gopal of SID, Manju and Ramesh are also to be mentioned. Mr. B M Rathnakar and Chandru have enabled me to shuttle between my house and the Lab in the last months.

All the facilities in the campus are used to the limit of their capacities. Most important amongst them are Hostel and Mess, courtesy Messrs. Sheshachala, Narayanappa, the mess and hostel attenders and many wardens. Nagaraj of Sudents' section also figures in the list.

The Campus Health Center was my second room in the campus. Dr. P H Prasad and Dr. Chikkalingaih treated me (treating me even now!) so well. Ms. Mary, Seenanna, Dr. Yogesh, Dr. Ghorpade and so many sisters in CHC have treated my Mental, Physical, and Dental ailments. They have kept my health intact to reach this point in my life, even, at the cost of being threatened by contracting infection from this *unhealthy* criminal!

Mr. Ravi and his boyz of Campus Xerox Center were involved in many of the *copyright* violations that were done during the course of this work. He is even responsible for the final copy that you have in your hands.

Sridhar *the atman*, Rama *the backseat driver*, and Kini *the extremist* tamed me with some work or the other and tried to bring in professionalism in all the works. While Sridhar brought in the (criminal!?) values in my thoughts and actions, Rama encouraged me to express my feelings courageously (later he only had to pacify me, when I overreacted). Kini showed me the way to do some work fearlessly. They kept my spirits always high without using *the spirit*. We were the *dushta chatushtayas* of *Prasthutha*. I never forget to implicate so many speakers we brought on *Prasthutha*'s platform who motivated me.

Our Registrar Dr. Udaya Balakrishnan, Asst. Reg. Mr. M. S. Venkatesh PRO Mr. Veera Raghavan and Security Officer Chandrashekhar gave me my memorable moments in the campus and encouraged my *guerrilla* activities. Now you know why I am in this unenviable position, i.e., writing a confession!

With each passage above I know that my part of the punishment is growing thinner!

The biggest of all the abettors are my parents who gave me all the freedom that one can have which is beyond constitutional. They kept on helping me, looking after my needs and demands. I should tell the truth that they don't actually know what I am doing. I have no choice; under oath I am disclosing everything. I feel sorry to say that I am spoilt because of them. Of late, my banker uncle Sathyanarayana Rao and my aunt Smt. Pushpa have started putting some economic sense into my (other)wise nut-head besides taking care of my parents and me. Vinay *the transporter* and Vijay *the programmer* have joined them. My brother (Dr.) Murali was so nice to me who constantly kept my criminal ideas flourishing by telling me how to publish papers!

Again, since I am under oath I confess that it is my brother, lawyer Subramanya's and his senior Haranahalli Ashok's scheme (very *schemy*, yeh?!) that I should confess and involve as many parties as possible (of course, to indirectly influence the adjudicator!) and tender a confession as an approver! Anyway I can't name them since they have the immunity for all the advice they give me as lawyers.

There are many others whom I should not leave; Dr. Udupa, Dr. Purushottam, Dr. Pandian, Dr. U J Shenoy, Dr. Santhosh Paul, Dr. M K Rajesh, Dr. Vidya Shankar, Dr. GNar, Dr. H N Nagendra, Dr. S Raghavan, Dr. Naren Naik, Dr. Sai Shankar, Dr. Purna Prajna, Dr. KK, (Dr.) Pai, (Dr.) Diwan, Hariharan and *Gandhi* Vishwa have all pulled me onto their path. I was allured[5] by their soft-spoken words. It was a sugar coated bitter pill that I swallowed then unknowingly.

Ramki, Maneesh, Dinesh, Srivathsa, Prasanna, Nandita, Pavan, Kishore, Sunil Kore and Dharmadhikari were my guru bhayees. Pramod, Sudar, Aditi & Ashwini Kanhere, Krupa, Deepak, Raamesh & Gayathri, *samsari* Suresha, *dhadiya* Ravi, Amod, *poli* CT, *manmatha guru* Prakash, *elasu* Bhatta, Vinay *with VD* and Biscuit Boyz, and many of my L-friends pepped up my spirits. Apart from these Smt. Vijaya Kanhere, and Raamesh's parents were no less in doing this. There were many others from students' council whom I don't enlist here (good for them!). Some used to just give *Gyan* whenever required. I was pampered by all of them beyond control leading to this situation wherein I am naming one by one. I am no way responsible for their actions. An unknown kid during Orissa cyclone relief activity and chirpy Ushnaa are minors to book them.

I beg your pardon. I wouldn't have done this work with my usual self. I was driven into it when I was sporadically normal!

I hereby declare that all my statements are correct to the best of my knowledge. I close my statement by telling that I did execute this work but under *duress*. If you want to punish, I am not the only one! Actually, it is to the contrary; I should be rewarded for telling *the truth*! I would like to be considered as an *approver* for prosecution and I should be exonerated. Now, tell me with so many parties involved does yours truly stand any conviction?

Epilogue

> "Man had always assumed that he was more intelligent than dolphins because he had achieved so much — the wheel, New York, Wars and so on, — whilst all the dolphins had ever done was muck about in the water having a good time. But conversely, the dolphins had always believed that they were far more intelligent than man — for precisely the same reasons." - Douglas Adams: The Hitchhiker's Guide to the Galaxy.

Dear smart dolphins! Phleeezzz kyp away from this dokyument! It's ment for less intelligent creations, even lesser mortals. U hav no nyd 4 konfrnc on VoIp!! I'm widh u.

[5] I can resist anything but *temptations!*

Statement

I hereby declare that the matter embodied in this thesis entitled **"A New Paradigm for Audio Conferencing on Voice over IP (VoIP)"** is the result of the investigations carried out by me in the department of Centre for Electronics Design and Technology, Indian Institute of Science, Bangalore, India, under the supervision of **Prof. H. S. Jamadagni**.

In keeping with the general practice of reporting scientific observations, due acknowledgement has been made whenever the work described is based on the findings of other investigators. Any omissions which might have occured by oversight or error in judgement is regretted.

R. Venkatesha Prasad

July 29, 2003

List of Acronyms

ALE	Application Layer Emulation
ALED	Adaptive Linear Energy-Based Detector
ALF	Application Layer Framing
ALM	Application Level Multicasting
APU	Audio Processing Unit
AS	Autonomous Systems
AVL Tree	**A**delson-**V**elskii and **L**andis Tree
B2BUA	Back-to-Back User Agent
BT	Background Traffic
CDF	Cumulative Density Function
CHI	Computer Human Interface
CMC	Computer Mediated Communication
CP	Call Processor
CS	Conference Server
CSCW	Computer Supported Cooperative Work
CSRC	Contributing Source(s)
CTI	Computer-Telephony Integration
CVAD	Comprehensive VAD
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DIVE	Distributed Interactive Virtual Environment
DN	Directory Number
DPM	Distributed Partial Mixing
FG	Floor Grant
FLOP	Floating Point Operations
FM	Frequency Modulation
GUI	Graphical User Interface

GSM	Global System for Mobile Communication
HTTP	Hypertext Transfer Protocol
ICQ	I Seek You (Chat and Internet Services Provider)
IETF	Internet Engineering Task force
ILP	Integer Linear Programming
IP	Internet Protocol
IRC	Internet Chat Relay
IRI	Interactive Remote Instruction
ISP	Internet Service Provider
ITU	International Telecommunication Union
LAN	Local Area Network
LED	Linear Energy-Based Detector
LP	Liner Programming
LPC	Linear Predictive Coding
LSED	Linear Sub-band Energy Detector
MASSIVE	Model, Architecture and System for Spatial Interaction in Virtual Environments
MBone	Multicast Backbone
MC	Multipoint Controller
MIPS	Million Instructions Per Second
MMC	Multiparty Multimedia Control
MOS	Mean Opinion Score
MP	Multipoint Processor
MCU	Multipoint Control Unit
MINT	Multimedia Internet Terminal
MTU	Maximum Transmission/Transfer Unit
NTP	Network Time Protocol
OOTW	Out of This World
PBX	Private Branch Exchange
PCM	Pulse Code Modulation
PM	Partial Mixing
PMM	Pattern Matching Multicast
PoCS	Proof of Concept Setup
POTS	Plain Old Telephone Sets
PS	Proxy Server

PSTN	Public Switched Telephone Network
QoS	Quality of Service
RAT	Robust Audio Tool
RFC	Request For Comments
RSVP	Resource Reservation Protocol
RTCP	RTP Control Protocol
RTP	Realtime Transport Protocol
SCCP	Simple Conference Control Protocol
SDES	Source Description
SDP	Session Description Protocol
SFD	Spectral Flatness Detector
SIP	Session Initiation Protocol
SIPS	SIP Server
SLA	Service Level Agreement
SNR	Signal to Noise Ratio
SSRC	Synchronization Source
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
UA	User Agent
UAC	User Agent Client
UAS	User Agent Server
UDP	User Datagram Protocol
VAD	Voice Activity Detection
vat	Visual Audio Tool
VMS	Voice Mail Servers
VoIP	Voice over IP
WAN	Wide Area Network
WFD	Weak Fricatives Detector
WWW	World Wide Web
ZCD	Zero Crossing Detector
ZCR	Zero Crossing Rate

Synopsis

Introduction

With the advancement and popularity of the Internet, the traditional circuit switched telephony applications are now implemented on this new medium. This is a major shift in paradigm from circuit to packet switching. This shift provides some advantages like higher throughput but also introduces a new set of problems in achieving the necessary quality in the Internet based implementations of circuit switched applications. A growing service in the packet switched paradigm is Voice over IP (VoIP) in which audio conferencing is an important and challenging feature. Therefore, there is a great interest in supporting this service over public Internet and Intranets.

There are many standards for audio conferencing on packet switched networks such as H.323 of International Telecommunication Union (ITU-T) and Session Initiation Protocol (SIP) of Internet Engineering Task Force (IETF). Visual Audio Tool (VAT) by Network Research Group of Lawrence Berkeley National Laboratory and Robust Audio Tool (RAT) from Network and Multimedia Research Group of University College London are examples of some of the earlier applications developed for VoIP services. All these standards and implementations either consider a small number of participants to support in a conference (for example, H.323) or assume that multicasting is available to support a large number of participants and further it is assumed that only a small number of participants (generally one, using appropriate floor control) would be speaking in the conference simultaneously. The new IETF draft recognises that multicasting is difficult to be enforced and maintained in the existing Internet. Therefore, there is a need to propose an architecture that limits the number of participants speaking simultaneously, but at the same time without tight control of the conference and which works in the existing Internet scenario with and without a multicast enabled network. When a large number of participants are to be supported, scalability is required. This thesis deals with important issues in providing the voice-only conference support on Voice over IP.

Contributions of the Thesis

This thesis proposes a new methodology for audio conferencing on VoIP. The features

of this proposal are; (a) smooth switching of active speakers in a conference without an explicit controller for turn-taking of participants (conference becomes tight and artificial, if there is explicit control), (b) scalable architecture for conferencing, with and without multicasting, (c) closeness to the real-life conference by providing good quality of conference in terms of participants' interaction, audio quality, and (d) reduced traffic compared to the conventional conferencing without multicast. Implementation of a total VoIP system with conferencing is also discussed.

To achieve the above, a distributed architecture is proposed for conferencing. This is a two level architecture with Clients (the front-end software for participants) at the lowest level connected at the next level to Call Processor for conference control and Conference Servers (called Selectors for reasons given in the thesis) for only handling audio packets. The Call Processor handles only the control messages such as participants joining or leaving the conference, starting or ending the conference, and maintaining the connectivity for the Clients and Selectors. The conference is controlled using only the audio packets by the Selectors. Selectors, distributed across the network, uniquely select the participants and limit the number of participants who are allowed to speak at any instant of time. There is no explicit floor control. Traffic in each user domain is aggregated at selectors and only selectors communicate with each other for conferencing. Therefore, multicast usually needed for communicating to multiple users can be replaced by unicast between Selectors when the network does not support multicast.

In an audio conference the voice quality decreases with the number of simultaneous speakers. Therefore, it is ideal to have only one participant speaking at any time for the best quality. But this lacks the interactivity of a conference since there is no provision for impromptu speech. A limit on the number of simultaneous speakers (N_{Max}) is specified - using a *conjecture* proposed here with the help of *Conversationa Analysis* - which assures interactivity as well as quality. The change over between speakers must also be smooth.

A quantifier called *Loudness Number* is defined that helps in selecting N_{Max} speakers who can speak in the conference simultaneously. The loudness number is defined in such a way that it avoids dropping a speaker abruptly and gives a fair chance for the other participants to speak in the conference. Loudness Number is also used to reduce the traffic on the Internet. A simple floor control mechanism using Loudness Number is presented to select speakers who are allowed to confer.

The traffic in this scheme is reduced because of limiting the number of simultaneous speakers to N_{Max} using the Loudness Number. Two heuristic algorithms that reduce

the traffic even further by exploiting the characteristics of Loudness Number are also presented.

Speech always has a few pauses, particularly in an interactive call when the speakers listen to each other. Sending packets that do not have useful audio content can be avoided when a speaker is silent. Voice activity detection (VAD) algorithms are introduced to do this. A study of the algorithms and their performances are discussed. VAD and Loudness Number together can be applied to induce further reduction in traffic.

The quality of speech depends considerably on the percentage of packet loss during transportation. The packets may be dropped in the network as well as at the playout buffers at the Selectors/clients due to the delay jitter. An analysis of VoIP packet flow on the Internet (between two nodes) is given. Simulation result is also presented to find the total percentage of packet drops during transportation and at the fixed size playout buffer for different packetisation intervals. The playout buffer is modelled using an M/D/1/N queue with vacations. A graphical representation of a case study of relation between the packet loss, delay and Mean Opinion Score (MOS) is presented.

The design and implementation of Selectors and the Call Processor are described. The Protocol used between the Selector and Call Processor that is used for conference control and setup is also described. Further, a correspondence between this design and the IETF's SIP standard is established. A method for handling media streams before mixing and/or selecting packets, in turn speakers in a conference at selectors is also discussed. A heuristic algorithm is given to address the conference server allocation problem that can be modelled as Facility Locator problem, which is an NP hard problem. A comparative study of how best this heuristic algorithm works is also covered.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

This thesis deals with a new paradigm for audio conferencing on the Internet with the main objective of providing the service that is difficult or not viable in the present network, using minimum resources such as bandwidth. It focuses on designing an efficient distributed audio[1] conferencing service on the Internet. Furthermore, the solution proposed here allows adequate number of simultaneous speakers and nearly unlimited listeners in a conference to impart a natural feel of a face-to-face real-life conference without an explicit floor control. A set of distributed servers to provide this application support is addressed, while managing the hurdles and limitations introduced by the present day Internet.

Traditionally, Public Switched Telephone Network (PSTN), a circuit switched network, has been used for voice communication. Even after packet switched data networks came into existence, circuit switched network has continued to be in use for voice communication. This is because data networks were designed and used originally for exchanging binary data between two computers and could not handle analog voice signal. Also, control protocol for data and voice connections are different. Progress in Digital Signal Processing and general purpose CPUs has made it possible to represent analog voice signal in digital form with limited bandwidth, and this can be handled easily by the data network. This has resulted in an overlap between the two networks for offering data as well as voice service. For example, data network is used for voice communication and PSTN offers Internet connectivity through modems. This service overlap has led to integration of data and voice services and a solution that works on the Internet, which is a packet switched network.

[1]The terms "Audio" and "Voice" are used interchangeably throughout this thesis. Henceforth, both the words mean speech unless otherwise explicitly stated.

The Internet has evolved as the most powerful and useful communication technology for communication and sharing information. It is the largest network and is still growing at a rapid pace. Progress in hardware technology has matched the ever demanding and resource hungry innovative software solutions. This has facilitated the use of Internet as a huge information store where information can be accessed quickly and with ease. Although there are many protocols through which computers can "talk" to each other, the Internet uses TCP/IP protocol suite. In TCP/IP, Internet Protocol (IP) is used as an inter-networking protocol. It is the most widely accepted and supported protocol for inter-networking across networks and terminals on different operating systems.

For voice communication using IP, binary audio data is divided into small chunks. These chunks are carried as payload in IP packets (typically one chunk per IP packet). IP packet header contains information about the source and the destination. The network is transparent to the type of the data it is carrying and these voice packets are nothing but data at the IP layer of the network. It is the task of the application that is used for receiving these packets to interpret and use it as audio packets. Once received, these audio chunks can be played at the destination. Although audio data is divided into small chunks for transmission, continuity of voice can be maintained if these small chunks are played in the proper sequence and in quick succession at the receiver.

The main investigation in this thesis is the transmission and reception of audio packets to provide an interactive conferencing service over an IP network, i.e., Voice over IP, popularly known as VoIP. Before discussing VoIP paradigm and the conference application dealt in this thesis, a brief look at the switching methods and the ideas behind packet switching would ameliorate the readability of this thesis. A brief account of the market projections are also given later to highlight the importance of VoIP technology in the current market.

1.2 Switching

In any communication that involves a large number of users, connection establishment before the actual data transfer is an important step. Switching is a term used to denote the establishment of a path for communication between a caller and a callee or sender and a receiver. In telephony, the term also includes other facilities such as exchange functionalities provided to the user. Moreover, the characteristics of the connection used for voice or data transmission is as important as connection establishment. Two

distinct switching paradigms are given below.

1.2.1 Circuit switching

Circuit switching is used predominantly for interactive communication like voice and video till now. Path establishment is based on extensive signalling and information interchange, using error free communication channel. Establishment of a connection is through a hierarchy of switches. Once the connection is established, it remains throughout the duration of the call and is dedicated to that call. The switches use Space Division or Time Division multiplexing. In space division switching, connection is established through galvanic connections. Actual switch connecting the users is called a cross point. In the case of Time Division switches, connection is established through data exchange in a memory. Connection between two parties is at a specific and dedicated time-slot. Actual switch is called a speech memory or time switch.

All the facilities offered to subscribers are through a centralized entity called an exchange. The facilities offered to subscribers include functionalities such as call transfer etc. These functionalities are built within the exchange. Tariff computation uses extensive monitoring of the signalling which is used for establishing calls.

1.2.2 Packet switching

Packet switching is widely used for non-interactive communication between data terminals and for computer communications till now. Path establishment is ad hoc and not permanent. The path is through intermediate hierarchical routers that switch each packet using destination address. An end-to-end connection is established rather than a single specific path between the source and destination. The connection may be on an error-prone communication channel. Connections between the routers are dynamic and they may change anytime depending on the link conditions.

It is predicted that packet switching would grow faster than circuit switching and all circuit switched applications/services would be realised on packet switched network. Phillips Infotech [1] and Intecom's white paper [2] show a sharp decline in circuit switched networks in this decade. In fact, the irony is that the services offered by telephone networks are supported on Internet. These services are implemented currently on circuit switched telephone network that enables the connection to the Internet, linking the end terminals or a subnet to the Internet Service Providers (ISPs). There is an ongoing debate (Isenberg [3, 4], Andrew Odlyzko [5, 6] and Vrsalovic [7]) whether the "intelligence" in a network should be put in the network or at the end systems. Unlike

a circuit switched network a packet switched internetwork, sometimes referred to as a "Stupid Network" [3, 4], has the least intelligence in the core. The debate is briefly presented below.

1.2.3 Intelligence: Where to implant? Network or end systems

The first form of telephony service was provided by manual exchanges where the intelligence of the system was the capability of the persons who enabled the facilities (though very limited) requested by the subscribers. The advent of technology has replaced manual intervention by more sophisticated systems at the central exchange. Intelligence here means switching capability, service functionalities, tariff calculations, guaranteeing Quality of Service (QoS) (which is a little easy with circuit switches), and managing the available resources efficiently. With this understanding of "intelligence", this section initiates a discussion on circuit switched versus packet switched networks.

Circuit switched networks usually have intelligence in their core, such as exchanges. Packet switching more often places intelligence in the end systems, as the end system is capable of handling a reasonable amount of computation. This debate as to where to place the intelligence is highlighted with Isenberg [3, 4] supporting the end system intelligence and Telecom companies pushing for centralized intelligent smart networks. A brief account of both the arguments is given below.

In recent years, the packet switched network has witnessed a huge growth and attracted many innovative applications such as Internet browsing and shared tools for collaborative works. The growth of Internet encourages convergence of many other media types such as realtime voice/video to be transported on the same network transporting data. It is observed that there has been up to several thousand-fold decline in key infrastructure costs over the last two decades. In the last twenty years, high growth rate in the volume of data traffic is recorded [3, 4]. The volume of data traffic is now overtaking the volume of voice traffic, which itself is growing, slowly though. Many different data types are now transported over the telephone network (despite the fact that the network is not optimised for all these data types). Many different types of "communications technologies", from television to Ethernet, that are not part of telephone network architecture have developed and grown in the recent past. The Internet makes the details of network operation irrelevant. It is hence shifting control to the end user.

Till late 1980's Telecommunication networks were handling dumb terminals at the edges (user premises). This pushed them to use the state-of-the-art switches and in-

telligent networks to support new services and upgradations. Indeed, Isenberg [3, 4] observes that in a world of dumb terminals and telephones, networks had to be smart. However, in a world of smart terminals, dumb networks would do. In a "Stupid Network", control passes from the centre to the edge, from the smart switches of telecom companies to users with an abundance of processing power at their fingertips. The centre of the network is based on plentiful infrastructure, inexpensive bandwidth and fast switching. A data object in a stupid network, like water in a river, gets to where it must go adaptively, with no intelligence and no features, using self-organising engineering principles, at virtually no cost [3, 4]. Data flows define the movements and channels within the system. Low cost of computation and infrastructure, under-specification, abundant infrastructure, and Internetworking characterise the stupid network. This allows users and application developers to freely develop new applications and innovative systems without actually requiring to know anything about the underlying network, even at the expense of efficiency. In fact, Isenberg overlooks the QoS requirements as he assumes that the bandwidth and computation are so inexpensive and abundant that the QoS is not worth worrying about!

Andrew Odlyzko [5, 6] takes a moderate view and cites an example of IBM[2] PC development that could not render the mainframes obsolete but gave an opportunity for faster development of newer, simpler, and progressively useful applications in large scales. His argument is, "Innovation is the Key" and the development of the killer applications (for example, *Mosaic* for browsing) with ease is the direction that the network would be taking. He opines that dumb Networks and smart Terminals are the ones that will attract many users/customers.

An investigation on the impact of the Internet on the main telecommunication architectures by Jean-Pierre Hubaux [8] found two trends. First, to keep the PSTN network as the only transport infrastructure, devolving a part of service and data logic on the Internet. Hubaux showed, with an implementation of Virtual Private Networks (VPN), that interoperation between any two Internetworks operated by different stakeholders could be achieved. The second trend is towards using the Internet to deploy telephony services. An observation of the new services shows this trend taking more shape at present. In his editorial [9] Jay Blickstein sums up the mood at present: "It's the users who enhance the network's functionality by inventing new applications, and anyone can be a player - not just the phone company".

There are some other voices [7] against those conjecturing the feasibility of such a network which gives full capability to the users, with the network itself being dumb.

[2]I think there's a world market for about five computers. - Thomas Watson, Founder of IBM.

The reasons are two fold (a) Network support is not free however affordable the cost of components may be. (b) Switching circuit cost is a fraction of overall cost of operating a network. Thus there is a need for a smart network that is reliable. With more and more disintegration of the network and the end terminals, the net efficiency is progressively falling. In addition, the available bandwidth will not be abundant as the requirement of bandwidth has grown many fold. The available bandwidth has remained a constraint in spite of the use of optical fibre links[3] in the last decade. Therefore telecom networks may start supporting the Internet reliably and the services are offered by ISPs through applications running on the end terminals.

A recent paper by Ben-Ameur and Herve Kerivin [10] discusses VPNs in detail. It is a general-purpose communication environment allowing user organisations to communicate privately, securely, and transparently through shared resources provided by telecommunication operators (like AT&T and France Telecom) that is, it emulates a private wide area network. It has a new telecommunication design methodology to take care of the uncertainty of traffic among corporate sites and develop economical virtual private networks. Service Level Agreements (SLA) are used to support QoS guarantees to the services. SLAs are contracts between customers and network operators that depend on the traffic characteristics of operations of users, including user demand, technology and competition. The forte of VPNs is their ability to use QoS routing [11]. Duffield et al. [12] discuss resource management in VPNs. The trend is to integrate all useful applications, at least for business/commercial/corporate offices - at one place - on PCs which can be safely and easily handled by ISPs and VPNs.

While this debate is continuing, this thesis does not take any stand on this issue. However, the growth of Internet is undeniable. Blumenthal [13] discusses in more detail on the design of Internet to aid its growth. The following are the main developments observed over the last decade.

1) The cost of the end terminals has continually reduced. Also, computation capabilities at end terminals are greatly increasing.

2) Internet is growing at an enormous pace, and many media can be used at the same time on it.

3) Internet poses many challenging problems and solving those would make the work place very efficient. For example, integration of different media

[3]The popular comment on memory hogging softwares and inefficient use of growth in technology: "What Andy Grove giveth, Bill Gates taketh away".

types for collaborative work increases efficiency of the work and the speed of inter-personal/group communication.

4) Opportunities for development of creative applications (including collaborative work with computer mediation and Computer Human Interaction (CHI)) are increasing with the growth of Internet.

5) Bandwidth and switching capacities are rapidly increasing.

6) Telecom services and Internet services are increasingly crossing each other's domain.

One important aspect of pursuing voice transmission on Internet is cost effectiveness. It is impossible to pronounce anything concrete about the cost of supporting audio and/or video on Internet compared with other technologies. Naive cost comparison is not useful because of the uncertain market and unpredictable market players who are setting/bringing up new technologies.

It is therefore highly desirable to develop and maintain some of the useful applications such as voice communication on the Internet using the computation capabilities of the end terminals. Intelligence and computing facilities at the terminals open up various opportunities for an enhanced application development. A good example is the campus-wide VoIP Private Branch Exchange (PBX) of Columbia university using Session Initiation Protocol (SIP) [14]. Hence it is time now for realtime services and data services to converge on the same network. Now is the for VoIP to *Festina Lente*, i.e., "Make haste, but cautiously".

1.3 VoIP

VoIP is still not the main technology in providing the voice services unlike circuit switched telecommunication technology. At present - with a few exceptions - it is more of a technology for the corporate offices and business establishments harnessing its effectiveness within organisations. However, it is observed that in the past few years it is picking up in the domain of ISPs supporting home network users. The advantages and limitations of the technology are taken up briefly here.

1.3.1 VoIP: The Plus and the Minus

The plus

VoIP technology consolidates the voice/data network infrastructure. This consolidation brings forth reduction in traffic because of the many coding techniques. The facilities provided can be easily moved to any type of customer premises. It allows easy addition of facilities and upgrading facilities provided to the customer. Multiplexing with other traffic consolidates inter-office voice traffic onto a wide-area network, replacing off-premises lines and tie lines provided by telecommunications companies. Thus the cost is reduced due to sharing of the link that is used for the existing data transportation. Further reduction is possible due to bandwidth saving by using low bit rate coding. IP network has some advantages, such as (i) it runs on a variety of networks (ATM, Frame relay, WLAN, etc.) and (ii) it is easy to monitor and administer. A summary of important benefits are paraphrased here from [15].

Compression:

Internet telephony allows parties to use the encoding most appropriate to their quality needs. For example, a user may decide for a call with lower cost or for toll quality. It can also be traded for a full FM radio quality voice with little regard for price when required; for example, by a radio station reporter.

Silence suppression:

Sending audio as packets makes it easy to suppress silence periods, further reducing bandwidth consumption, particularly in a multi-party conference or in voice announcement systems.

Traffic separation:

Sending Fax across a circuit-switched network is rather inappropriate, as this is a delay-insensitive but loss-sensitive traffic. Thus facsimile traffic can be separated from voice traffic as close to the fax machine as possible and converted into either email messages or a TCP connection.

User identification:

Realtime Transport Protocol (RTP) [16] used for Internet telephony easily supports speaker identification in both multicast and bridged configurations and can convey more detailed information if the caller desires, compared to Caller-ID in PSTN.

Computer-telephony integration:

Computer-telephony integration (CTI) is very complex in conventional systems because of the complete separation of data and control paths and the separation of phone equipment from the PC's controlling them. Much of the call handling functionality can be easily accomplished once the data and control path pass through intelligent network-connected end systems.

Shared facilities:

Many corporations and universities already have high-speed local area networks. Given its low bit rate, packet voice and low-bit-rate video can be readily supported on a well-designed (switched) LAN, even without explicit QoS support.

Advanced services:

From initial experiences and protocols, it appears to be far simpler to develop and deploy advanced telephony services like encryption in a packet-switched environment than in the PSTN.

Multimedia:

Adding media such as video, shared whiteboards or shared applications is easier in the Internet environment compared to the POTS and ISDN, as multiplexing is natural for packet networks.

User interface:

User Interfaces enable easy execution of many PBX and other services.

Separation of voice and control flow:

In telephony even though the signalling flow is carried on a separate network the flow has to "touch" all the intermediate switches to set up the circuit. Since packet forwarding in the Internet requires no setup, Internet call control can concentrate on the call functionality rather than on connection. For example, it is easy to avoid triangle-routing when forwarding or transferring calls; the transferring party can simply inform the transferred party of the address of the transferred-to party, and the two can contact each other directly. As there is no network connection state, none needs to be torn down.

Feature ubiquity:

The current phone system offers many different features depending on whether the parties are connected by the same PBX or reside within

> the same local calling area or are connected by a long-distance carrier. Facilities provided by one vendor may not be feasible when a call is supported by more than two different systems. For example PBX and PSTN service such as 'Call Forwarding' may not span across them. Internet telephony does not suffer from this problem. This is because Internet protocols are internationally used and because services are defined largely by the end systems.

Further, it allows easier integration of voice for call centres and other applications. The viewpoint of this thesis is: The ability of VoIP to integrate voice with many other applications/services like white board, video, and sharing of data for collaborative work over large geographical distances is the most attractive aspect. There is also scope to provide value added services at lower cost in the end terminal, as it is intelligent unlike a dumb telephone set in the PSTN. User interface (customer premises equipment in telephony jargon) development for handling new services - as and when added - is easily possible in VoIP.

The minus

Realtime traffic is transported on packet switches. Therefore there is an overhead at the receiver, as it has to assemble voice segments in various packets streamed and play it out. Packet transportation results in a delay compared to circuit switched dedicated connections. Furthermore, the delay is not constant for every packet that is transported, as the Internet operates on a best-effort basis at present and cannot guarantee an upper bound on end-to-end delay or lower bound on bandwidth. Thus the receiver must collect packets and assemble them properly and ensure that there is a continuous playout. Lost audio packets are not re-transmitted to preserve the realtime feature. The applications have to face this eventuality of packet loss to the extent possible. Some lost packets may be recovered if redundancy is introduced. Coding packetised speech with lower bandwidth codecs when bandwidth is not sufficient may also reduce packet loss. All these factors affect QoS and are dealt in detail in [17].

The direction of the future of Internet technology (though Internet Protocol (IP) is ubiquitous now) is uncertain. Nonetheless, there are some limitations for VoIP technology to be installed and run in every subnet. The installation costs, it is observed, are the same or more than those for PBX. These systems still lack many features found in PBXs because of incoherent technology growth and R&D by various application developers. Analog integration can be problematic in some cases and the existing solutions are not yet fully evolved. Standards for powering phones over Ethernet are

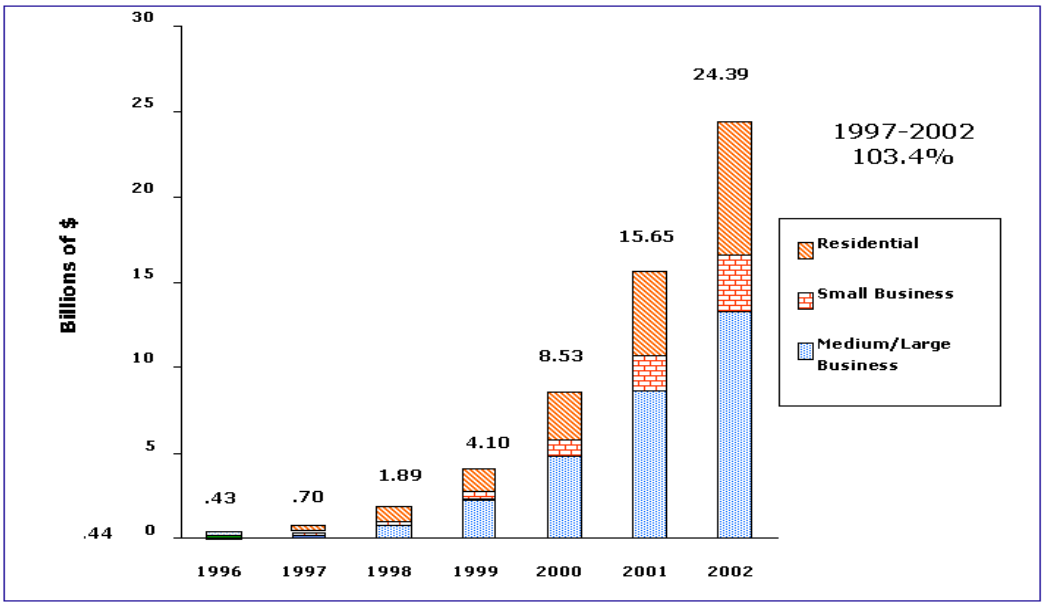


Figure 1.1: **Forecast and assessment of VoIP market till 2002**

still evolving. Guaranteeing QoS, that is an essential part of realtime services, is not presently fully available to the end user.

1.3.2 What does the market say?

The research work considered in this thesis has direct relevance to the VoIP market. VoIP technology from the business viewpoint is considered here very briefly. IP telephony is an evolving market that has potential for tremendous growth in the telecom sector. In fact, IDC, a market research company, goes to the extent of saying that the 'market for IP Telephony will weather the telecom storm' [18]. Table 1.1 shows the huge market for IP telephony Equipment revenues. Figure 1.1 shows the assessment of total market value of packetised voice services till 2002. A market research [19] shows a overall growth of Internet telephony to $4.2 billion by 2006. The safe bet in telecommunications now is IP telephony since, its market is evolving at a rate higher than ever before. Some insights and projections are also in [20].

1.4 Teleconferencing: Motivation for this Work

This thesis is on VoIP Conferencing. This section traces some of the advantages of a distributed conferencing (not necessarily just over Internet). *Telepresence* [21] is

Table 1.1: **Worldwide IP telephony equipment market revenue Assessment and Forecast**

Year	**2001**	**2002**	**2003**	**2004**	**2005**	**2006**	**2007**
Total $M	1846	2398	3336	4945	7592	11356	15142
Growth(%)	NA	30%	39%	48%	54%	50%	33%

Source: IDC January, 2003

the enabling technology of human interaction at a distance[3], creating a sense of being *present* at a remote location. Telephone is a well-established telepresence service that extends human speech and hearing. Moreover, voice is an important medium in any collaborative work. Thus audio conferencing enables many to be *present* with others at the same time.

The costs of meetings, travel, collaborative work, and entertainment have spiralled upward without the use of network. On the other hand, the costs of virtually all types of video and teleconferencing systems have progressively dipped, and the applications have become much user friendly. At present collaborative works without the use of some form of (circuit/packet) network is unimaginable. In the last few years, teleconferencing hardware and software innovations have been major areas for developers and researchers. Systems today use high technology, are easy to operate and are affordable. In many cases, the investment on teleconferencing and videoconferencing systems is amortised from the savings realised with just the first use! Today, video and/or teleconferencing capability is at the top of the list when business houses look for ways to stay up with competition and to cut expenses.

The meetings of tomorrow will be conducted on several widely dispersed terminals, using a low cost, productive, and highly effective conference connected through the Internet. In many cases, the Internet will become the meeting room for teleconferences, big and small, as companies realise increased productivity and bottom line savings from productive meetings without cross-country travels. On-line multimedia presentations

[3]The reasonable man adapts himself to the world; the unreasonable one persists in trying to adapt the world to himself. Therefore all progress depends on the unreasonable man. - G. B. Shaw.

are viewable at hundreds of locations at the same time; instantaneous feedback and collaboration on documents and database are possible. Immediate decision-making, policy-making capability and empowerment, reductions in travel-related stress on staff and no travel time make work/business efficient.

On one hand the ubiquitous use of browser technologies, the increasing sophistication of multipoint teleconferencing options and growing cooperation among vendors in developing meaningful standards are fuelling the research on these services. On the other hand, advent of new technology that allows the industry to meet customer demands for better and faster delivery of realtime conferencing capability is also propelling installation of *the new* solutions. These solutions are creating new ways for people in business, academia, medicine and other fields to meet electronically, as they never ever did before.

Experts feel that ongoing concerns about cost-containment in business, education and healthcare - as well as demands for improvements in training - will drive an ever increasing interest in the benefits of video and teleconferencing technology - and the products will eventually follow [22]. Collectively, audio conferencing on network is cost effective, consistent (same data-voice-for all), exciting, faster, involved (as the interaction is immediate), ubiquitous [23], and it enables long distant meeting.

Existing telecom companies can provide video and/or audio conferencing solutions. For example, *Telepresence Wall* was built by France Telecom using constant connection between two locations [20]. However, the facilities to the user can never be comparable when the same service is provided on the network (Internet). The network can integrate all other media of collaborative works like file sharing, video, white board and private chat. A large number of participants geographically far apart are reachable on Internet using multicast/broadcast which is very simple compared to the circuit switched conference provider scenarios. Thus the use of Internet for conferencing would result in the integration of many collaborative media. Though it appears to many that realtime video over Internet for collaborative applications enhances the quality of interactions, Eckehard Doerry argues to the contrary [24]. Thus this thesis focuses only on the problem of audio-conferencing on Internet which is an important realtime medium of any collaborative distributed work.

1.5 Objectives

The goal of this thesis is to investigate a *distributed* architecture for audio conferencing that preserves sufficient interactivity and acts as a facilitator without requiring an

explicit or static floor control. The conferencing is required to closely emulate a *face-to-face* conferencing with adequate *impromptu* interaction. Packet traffic reduction and ease in implementation are additional requirements. This thesis sets out to bring out the issues involved in developing a multiparty conferencing application on Internet (IP networks) and provide suitable solution. The framework in which the solution is cast is given below briefly.

In a collaborative work voice conferencing is an important feature. It is assumed that the bandwidth for teleconferencing is available to the end terminals that are willing to participate. The concern here is to *design* an interactive conference application over the network, which uses the network bandwidth efficiently. A study of the conferencing application development on the Internet connecting many corporate networks together to support is attempted. Presently the traffic and the link occupancy by a type of service (say, WEB access on port 80) can be controlled in corporate networks to allow streaming [25]. This is not valid throughout the Internet. There may be some distortions occasionally; nonetheless, the conference service can still be supported since traffic on the link connecting to Internet is controlled. It is impossible to sustain a good quality service without any traffic control on links when the users sometimes start using the link for other services.

Another concern here is the scalability of the service when there are many simultaneous speakers in a conference. No assumption is to be made with respect to availability of multicast support by the network. If available it would be useful to make the system to scale to higher conference sizes on Internet. The solution, however, should work within this limit to enable the conferencing service using the multicast support if available. Providing interactivity, bandwidth reduction, ease of implementation and maintenance of service are other important concerns of the proposed solution. The solution is expected to be better for corporate networks compared to many available solutions. Since there can be some control over the traffic on the links of a corporate network maintenance of the solution is easy and straightforward.

Conferencing solution on circuit switched networks using conference bridges is expensive as it needs dedicated connections and is therefore not scalable. Existing packet switched conference solutions have some limitations. Thus this thesis should aspire towards building a fairly scalable audio conferencing. There is a need to have a fresh look at the VoIP architectures to cater to participants spread over a large geographic area taking into account the non-availability of multicasting. Further, there is a need for central and/or distributed controllers to manage the signalling part of the architecture.

There is also a need for dynamically managing a number of audio channels (Floors)

sufficient to render an interactive conferencing. When many participants are vying for speaking in the conference, there is a need for controlling the access to the limited floors but without compromising the interactivity of the application. Essentially, it is desired to dynamically decide who is going to speak next and who can interrupt the present speaker. The quality of any collaborative work depends also on the progress made in providing interactivity apart from the essential technical issues – data loss, bandwidth, etc.

There are many insights that are useful in enhancing the quality of the conference, by studying the packet dynamics vis-à-vis size of the playout buffers, packet sizes, etc. Therefore packet behaviour at the servers or at the participants' terminals is an important aspect that requires attention.

Reducing realtime traffic has a positive effect on the application scalability. Hence techniques for traffic reduction should be addressed using architectural properties and speech characteristics. Thus the development of algorithms (may be specific to architecture) to reduce traffic – and in turn the number of packets on network – is essential. The challenge is to find appropriate methods that conserve speech quality and require less computation while reducing the traffic on the network.

Also, the proposed solution should be easy to implement. The protocol for the new architecture and its conformance with respect to the standards in the field (say, SIP for VoIP) are important aspects. Optimising the allocation of servers to entities of the architecture is an integral part of the smooth conduct of the service, and this merits attention. An in depth study of many of the above *issues* concerning this service is given in Chapter 2.

1.6 Contributions of the Thesis

a) A new architecture that preserves sufficient interactivity of an audio conference.

b) Arriving at a limit on the number of floors for an audio conference for sufficient interactivity.

c) A new quantifier called "Loudness Number" for quantifying the participants based on their past activity for smooth speaker change with fairness to the present speaker and the participant who is trying to speak/interrupt. It is used to impart a natural feel of a face-to-face real-life conference.

d) Analysis of the audio packet behaviour on Internet, based on a simulation model.

e) Reduction of traffic using loudness number.

f) Voice Activity Detection (VAD) for reducing the traffic.

g) A simple conferencing protocol.

h) A unification of this protocol with SIP standard.

i) Software design of the "Proof of Concept Setup (PoCS)" for conference solution.

j) A heuristic algorithm for the allocation of conference servers.

Some of the contributions of this thesis are published as papers and are listed in List of Publications.

Why[4] and *how*[5] the above list forms the basis of a new conferencing paradigm is the focus of this thesis.

1.7 Organisation of the Thesis

Thesis is organised as follows.

Chapter 2 deals with important issues that need to be considered for providing conferencing support on network, in particular VoIP. Many existing solutions, standards and techniques are seen in the light of these issues. A comparison of them and the view taken by this thesis is given.

Chapter 3 starts off with a set of requirements for a VoIP conference. An architecture driven by these requirements and the components of the architecture are discussed. Features, such as scalability, of the new architecture are covered.

[4]The answer to the question as to why an issue is examined is inspired by the new requirements that often surface or is triggered off due to the related earlier literature.

[5]It refers to discussions on the path chosen for attacking an issue and novelty in the solutions, if any.

Chapter 4 addresses the use of conversational analysis to propose a conjecture that bounds the number of simultaneous speakers (number of floors) in a voice only conference. The chapter covers the background material of a model of conversation. Some comments and discussion are provided on how floors can be shared.

Chapter 5 proposes a new quantifier called Loudness Number, that can be used to select a set of speakers dynamically without blocking the current speaker unnecessarily. Comments on the Loudness Number and using it with the proposed architecture are also covered.

Chapter 6 answers questions on bandwidth saving in the proposed solution. Two heuristic algorithms are presented to save bandwidth. Voice Activity Detection (VAD) algorithms are also covered in this chapter.

Chapter 7 contains some simulation studies, and experiments on VoIP conferencing test-bed developed. Effect of delay and packet loss is a subject of major discussion. A simple simulation model to find packet loss in the network has been proposed. Comments on improvements in quality are made.

Chapter 9 concludes and summarises the work presented in this thesis. Criticism and author's view on the problems tackled are given. Possible directions for future explorations on the basis of this work are an important part of the chapter.

Each chapter brings about the motivation for the immediate discussion, builds the framework for the problem(s) tackled wherever necessary and concludes after outlining the limitations.

Chapter 2

On Issues, Techniques, Standards and Implementations of VoIP Conferences

> "...man will occasionally stumble over the truth, but usually manages to pick himself up, walk over or around it, and carry on." - Winston Churchill

2.1 Introduction: Collaborative Technology

The idea that collaborative technology is an activity in search of a need should be laid to rest [26] (Section 1.4). Collaborative technology's niche is among individuals who spend most of their time in group endeavours, who use computers to do their work, and whose potential for collaborations has been impaired by inadequate geographic proximity[1]. It seems especially well suited for the kinds of scientific collaborations envisioned for the "Collaboratory" (the collaborative laboratory). The trend among multimedia conferencing systems has been to draw on real world interaction protocols, but not necessarily on ensuring an honest-to-goodness an electronic replica of a face-to-face conference (except in studies by Redenkovic [27, 28, 29]). Multimedia conferencing is promoted as a supplement to face-to-face collaboration [26]. The desirable features of collaborative technology include: (a) less cognitive overload for the participants; (b) simplicity; (c) realtime media; and (d) good quality [26].

A realtime collaborative work demands faster interactions among the participants, immediate feedback/corrections for the speakers and mixing of the voice streams in realtime. These demands, if satisfied, result in faster decisions on the working subjects.

[1] If, as it is said to be not unlikely in the near future, the principle of sight is applied to the telephone as well as that of sound, earth will be in truth a paradise, and distance will lose its enchantment by being abolished altogether. - Arthur Strand, 1898

Collaborative works include pictures, whiteboard, file sharing, text and audio/video. It is often assumed that the different media types are controlled, supported and loosely coupled on a collaborative platform. Moreover, depending on the demand, availability and cost-effectiveness of services, these media can be included incrementally. For example, for less critical work only text conference (chat) is included and it can be supported on a network with a small bandwidth. Subsequently, based on the availability of bandwidth, whiteboard, audio and video may be supported. Inclusion of realtime video requires audio and lip synchronisation. Thus a loosely coupled collaborative platform should be able to integrate all the media types that are supported individually.

Media types such as pictures, whiteboard, and text can be easily included in a collaborative tool. However, audio and video are difficult to support. Audio is the most important component of any realtime collaborative application. It often carries the core content of the interactions and is an important component of a successful collaboration. Hindmarsh et al. [30], show that participants fall back on audio to resolve difficulties with other aspects of collaborative applications such as negotiating a shared perspective. In fact, Eck Deorry [24] demonstrates very marginal improvements in collaborative-ness when video is included. Hence this thesis focuses on the *Audio* medium of a collaborative work platform, i.e., on *Audio Conferencing*.

Design of audio for collaborative interfaces is complicated by constraints imposed by the environment, bandwidth requirement, realtime requirement and a large number of simultaneous participants. The underlying network requirements for any collaborative audio are very stringent (more stringent than for a non-interactive audio). Collaborative audio is sensitive to packet loss, latency and delay jitter, and to gaps in the audio caused by the lack of the realtime support on host machines. These factors affect intelligibility of audio and speech. There are two types of Audio conferences – non-interactive and interactive – that merit a discussion here.

2.1.1 Non-interactive conference

In a broadcast scenario or a panel discussion with open microphone only one person or a panellist would be speaking while the majority of the attendees are only listening. The set of active participants will not change frequently, or if it changes then it is decided by a moderator. For example, panellists may take turns sequentially to address the audience. These applications are, therefore, not interactive; they have mild requirements on the audio latency and stringent requirements on audio quality.

Even in semi-interactive applications such as remote lectures (for example, Instruction On Demand [31]) interactivity is moderate. A Participant can notify the speaker

that he has a question; the speaker may then allow a question, upon which the speaker and the questioner may interact live over the network. In this scheme the active participants are not predetermined but there is a control over who will speak at any given time. Such scenarios are not very interactive and are similar to the completely static case in the sense of negligible packet loss and higher tolerable delay.

2.1.2 Interactive conference

In an interactive collaborative work, participants directly interact with each other. The delay should be low here to accommodate interactivity in realtime. The set of speakers is not static. Simultaneous speech streams are to be supported. Then the delay between two streams may severely affect interactivity as human ear can easily recognise delays in excess of 150ms. In fact, ITU-T G.114 [32] standard specifies 150ms as the limit which is non-recognisable. When the participants are aware of delay, the upper limit is 400ms. Thus interactivity is delay sensitive. The major thrust in this thesis is on an interactive conference. After all, by supporting interactive voice, non-interactive voice conference such as broadcast type can be easily supported on the same platform.

Hitherto realtime interactive audio conferences were supported on circuit switched networks. More recently, standards and implementations for audio/video conferencing on packet switched Internet are available. There are many issues which do not arise in conference implementations on circuit switched networks, but they come into picture in packet switched networks. Therefore a comparative study of conference on circuit switched network and packet switched network conference solutions with a broader perspective is desirable and fruitful.

In this chapter, the concepts of conference in circuit switched implementation are presented briefly. An in-depth discussion of the critical issues in providing a conference service on Internet follows. With these issues in the background, two standards H.323 and SIP for VoIP call/conference are considered. This study leads to issues to be investigated in this thesis. As the standards are comprehensive for telephony, major changes to them are not permissible in the proposals of this thesis. More work may be required as these standards are still evolving. Some of the ideas in this thesis suggest refinement of these standards. Later a few existing implementations with or without these standards are considered. The discussion in this chapter marks a baseline, a benchmark[2] for further work carried out in this thesis.

[2]The difficulty lies, not in the new ideas, but in escaping the old ones, which ramify, for those brought up as most of us have been, into every corner of our minds. - John Maynard Keynes

2.2 Audio Conference on Circuit Switched Networks

The process of combining multiple analog signals to and from a conference call as multiple extensions on a single line is accomplished by merely bridging the wire pairs together to superimpose all signals. It is equivalent to parallel telephone connections.

When digitised voice signals are to be combined to form a conference they have to be either routed through a digital *conference bridge* [33] or they must be converted to analog form in order to add on an analog bridge. The digital conference bridge selectively adds the signals and routes the separate sums back to different subscribers. Usually PBX switching boxes have limited conference capability. Digital phones which are easy to implement require a conference bridge. In many digital switches the digital signals are converted to analog form, added and then reconverted to digital form. Some conference bridges add the digital signals after converting the companded PCM stream to linear PCM and then reconvert to companded signals. Instead of directly summing all signals, a refined technique is to switch only the loudest speaker by monitoring the activity. This technique allows for more participants in the conference because idle channel noise does not get added to the output [33].

As may be seen from the above description, these solutions work in a centralized architecture with central switches capable of handling many streams or bridges and with multiple line terminations. These solutions cannot scale as the hardware becomes the bottleneck. However, the delay, delay jitter or loss is never an issue in this hard-wired setup. More details on conferencing implementations and algorithms using digital switches are given in [34] and [35].

An example

Chelston Conference Bridge [36] is an example of a commercially available conference bridge service. The bridge can support only ten participants in a conference though a large number of lines (2400) can be connected to the bridge. All other facilities such as voting can be provided because this solution is centralized. However it is hard to scale. Intelligence is built into the core of the conference bridge.

2.3 Issues in an Interactive Audio Conference

Following are the issues in an interactive audio conference and are the basis for the evolution of the proposed standards, techniques and existing implementations. The design of an application should take into account the impact of these issues

I. Quality of the conference is based on interactivity, i.e., participants are to be able to freely acquire an opportunity to speak to other participants.

II. Mixing and customised mixing of speech for the participants.

III. Large scale distribution of participants over a wide area (scalability).

IV. Availability of multicasting.

V. Traffic reduction.

VI. Quality of the conference dependent on packet delay, packet loss and delay jitter.

These issues are explained briefly in the following paragraphs.

2.3.1 Interactivity

One of the main features in an interactive conference (Section 2.1.2) is to allow multiple speakers at any time without any control on who can speak, what can be said, and who can hear it. Gaming and other virtual reality services [37] may allow audio from every participant. There is already some work that allows each participant to speak without any restriction [27, 38, 28, 29, 39, 40, 41, 42]. But there are some issues that would make the scheme of allowing everyone to speak concurrently unwieldy - at least in a voice conference. This debate is taken up in the Chapter 4. This method of allowing every one in a conference is discussed in detail in Section 2.4.5.2.

Supporting most natural audio communication between participants allows spontaneity in the way people interact using audio medium. A case in point is a face-to-face conference wherein all sounds from participants are heard at all times. The fact that conferencing is as much about human factors as it is about underlying technology can never be over emphasised. It is crucial to devise collaborative systems with an understanding of how sociological and psychological factors impact group work, especially since mismatched expectations in group-oriented systems have resulted in serious groupware failures [43].

Floor is a virtual entity that allows a participant to speak. With some form of *floor controller* [44, 45, 46, 47] explicitly telling the participants when to speak and also reserving/waiting for one's turn to speak would evidently result in "gagging" of participants [27, 38, 28]. This means that the system's view of "speaking" is not the same as that of the participants. In addition it even hinders the natural urge to interrupt. This

makes a static and explicit floor control strategy [44, 45, 48] inappropriate in supporting most natural audio communication. Mixing speech from all participants as soon as they occur facilitates communicating the concerns of listeners to the current speaker. Without this capability collaborative platform would be imposing an unnatural curb on the participants. Thus the requirement of impromptu speech without explicit permission from any controller (deciding who should speak when) is a necessity to mimic a real-life conference. It may not be necessary or, for that matter, even desirable, to transmit all speech streams.

2.3.2 Customised mixing

Supporting customised mixing refers to supporting individual choice of weights for different voice streams at each participant's end terminal. This requires independent audio streams at the participant's end terminal. More specifically, this relates to providing support for individual spatialisation. Spatialisation refers to the control of a listener over the audio streams that are received before mixing. This capability is at the cost of having to transport all voice packets individually, i.e., unmixed, to all participants.

Hendrix [49] considers the presence and absence of spatialised sound and addition of spatialised versus non-spatialised sound to a stereoscopic display. It is reported that addition of spatialised sound does significantly increase one's sense of presence in the virtual environment. Addition of spatialised sound does not enhance the feel of realism of that environment though.

In a recent study, Watson and Sasse [50] use packet loss rates (typically 5% and an upper limit of 20%), microphone quality, volume differences and echo from open microphone for the measurement of audio quality of a conference. They vary volume and feedback from echo. Their experimental study includes: (1) *Quiet speech* – one stream recorded with low volume and other with normal volume; and (2) *Loud Speech* – one stream recorded with high volume and another at the normal level. Their result has shown some interesting aspects on the dependency of speech quality other than the loss percentage of transmitted packets. They conclude that when packet loss repair method, such as packet repetition, is in place, the audio quality degradation depends also on volume settings, microphone settings and echo. Many of these can be easily preset. Volume changes and different volume settings are harder to set unless some method is in place, and they affect the conference quality significantly. Thus support for individual customised mixing of audio at the participants' terminals is desirable as it provides participants an opportunity to enhance those audio streams that are of

more importance to them and to remove those which are disturbing or not important.

2.3.3 Large scale distribution of participants over a wide area (Scalability)

Scalability is much more important now than before as collaborative work requirements are growing across the globe. The issue here besides the large number of participants is the adaptability of many types of network connections and the end terminals including Operating Systems.

Schooler [51] identifies many issues regarding large scale conferences – (1) session models and its protocols; (2) multicast address management; (3) techniques for bandwidth reduction; and (d) codification of heterogeneity based on voice quality. Schooler [52] also identifies the ranges for multimedia conferences with respect to operating principles. A tightly coupled conference has up to 100 participants where as a loosely coupled conference can scale up to 10^5 participants. More than 10^5 participants corresponds to broadcast. Handley, et al., [53] identify session scaling based on one or more of the above aspects including security and authentication besides network support and reservations.

The key to managing heterogeneity over unmanaged, large scale networks lies in fashioning *distributed*, rather than *centralized*, solutions. Centralized solutions are easy to build and maintain [54]. They may not be adequate when the participants are geographically far apart because of poor response for time-sensitive operations as well as imposing heavier levels of network traffic in some areas. Distributed solutions that deal with local parts of the network are more suited to large scale setting of the Internet. Evidently, it is difficult to implement and manage distributed solutions.

2.3.3.1 Components

Design of large scale services across a vast geographical setting has to consider the specification from two perspectives:

(i) End terminal: Network connection, CPUs, I/O devices, storage capabilities, codec (dedicated hardware and software), communication protocol and network interfaces of the participants place limits on the end-system capabilities to process, consume and generate multimedia data. For example, limited resources in terminals can result in buffer overflow, delay in processing data and inability to process data. These manifest to the user as unacceptable playout delays, lost audio segments and poor user inter-

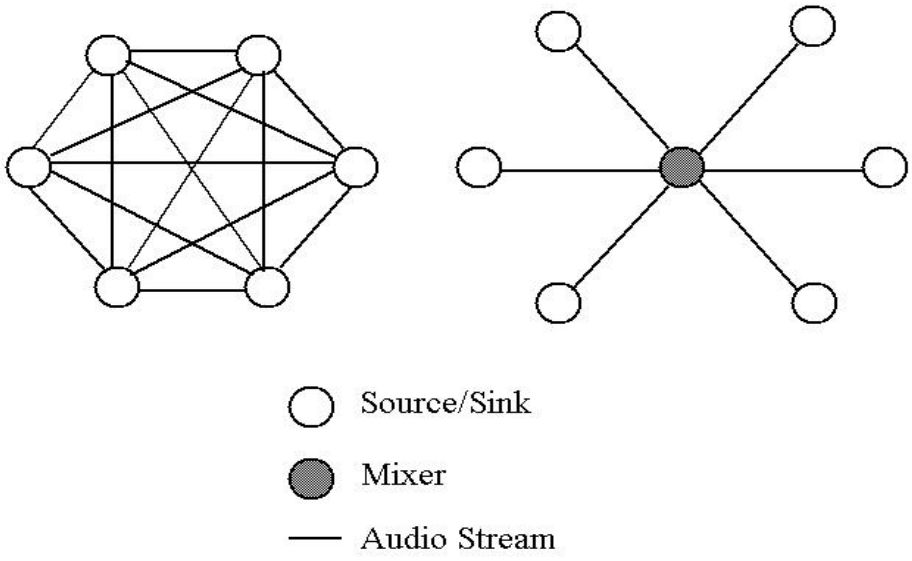


Figure 2.1: **Peer-to-peer and centralized server architectures**

action. Thus no assumptions regarding the capability of end terminals are advisable except a few minimal requirements.

(ii) Network: End terminals are likely to be placed in different networks with different bandwidth capabilities and protocols. For example, maximum transmission unit (MTU) may be very different for different networks. The use of Internet Protocol (IP) has made it simple to transact across many networks. It is assumed that the end terminals have TCP/IP and UDP/IP stacks on which the VoIP application runs.

2.3.3.2 Mixing of audio streams: At server or the end terminals?

The large scale implementations have to address the question of mixing of audio streams.

Peer-to-peer: When all the participants exchange packets, mixing can be done at the individual end terminals, thus allowing customised mixing for participants. Receiving independent streams would allow each listener to create personalised audio mix. This can be tailored to the individual's equipment, can be spatialised according to specific location within a shared virtual space and can be under individual control. The number

of packets in the network would be enormous when there are too many participants without any control (see Figure 2.1). Even with multicast, it is very demanding on network resources, particularly bandwidth, apart from computational resources at the end terminals.

Server mixing: When end terminals are not capable of handling many streams, introducing a server for mixing reduces the load on the end terminals as also the number of audio streams in the network. Traffic on the server network will be significantly high and most often it is cause of the bottleneck. Server mixing does not allow customised mixing for the end terminals. Thus interactivity would be reduced (Figure 2.1 [41]).

There can be many architectures slightly different from the above. Jorg Roth and Claus Unger [55] identify the following architectures for conferencing.

a) Architectures with centralized components: These have at least one centralized component. Peers are not connected to each other.

b) Architectures with direct communication: These have no centralized component at all. All peers are connected to each other.

c) Hybrid architectures: These have at least one centralized component and allow direct communication between peers.

d) Asymmetrical structures: These have no centralized component and distribution of components among peers is not symmetrical, i.e., at least one peer component has no replica.

e) Multiple servers: These use more than one server; i.e., centralized components are distributed to more than one site.

Support for individual tailored mixing and independent audio streams at the end terminals are offered by peer-to-peer transactions at the cost of network resources. Any change in the number of streams in the network will have a direct impact on the flexibility the end terminals have when creating customised mix. Server mixing offers efficient distribution of audio streams across the network. Therefore the preferred solution tends towards server mixing and multiple servers for large scale implementations.

2.3.4 Availability of multicasting

One of the main advantages of packet networks over circuit switched networks is the availability of multicasting/broadcasting. A single multimedia stream can be distributed to a large number of participants called subscribers. Deering's monumental

works [56, 57, 58] on multicasting offers an efficient multipoint delivery mechanism. A single packet is sent to an arbitrary number of receivers by replicating packets only at the forks of the network. Transmissions from one to many are accomplished without packet duplication within a subnet using a common group address. Multicast is based on the formation of groups whereby a number of processes may join a multicast group and will then receive all data sent to that group address. With the support of multicasting in IP networks, a participant may join as a member of the conference after bearing an allocated Class-D IP address and start sending/receiving packets on a multicast address.

There mainly are two issues at present with respect to multicasting: (i) Multicasting address allocations using IGMP [59]; and (ii) Enabling routers to allow multicasting across firewalls. These have not yet been resolved in the Internet.

Unfortunately, majority of the routers on the Internet today are not multicast-enabled. Although the number of routers equipped to handle multicast is growing, such products are still in a minority. Most routers are set up to move traditional Internet Protocol (IP) unicast packets – information that has a single, specific destination address. Router manufacturers have been reluctant to create equipment that can multicast until there is a proven need for such equipment. As one might expect, it is difficult for users to try out a technology until they have a way to use it. Without the right routers, there is no multicasting. Without multicasting, these right routers will not see the light of the day. A Catch-22 [60] situation indeed!

The experimental Multicast Backbone (MBone) set off in 1992 by Internet Engineering Task Force (IETF) raised many hopes to resolve this situation by producing more evidence for multicasting applications. It succeeded too. MBone is a "virtual network" – a network that runs on top of the Internet – with software that allows multicast packets to traverse the Net. However, MBone allows multicast packets to travel through routers that are set up to handle only unicast traffic. The software that utilises the MBone hides the multicast packets in traditional unicast packets so that unicast routers can handle the information. The scheme of moving multicast packets by putting them in regular unicast packets is called *tunnelling*. In future, Internet community hopes that most commercial routers will support multicasting, eliminating the affliction of tunnelling information through unicast routers [60]. An IETF draft [61] by Rosenberg and Schulzrinne of MMUSIC group comments:

> "...Unfortunately, multicast has not been widely deployed across backbones (some do, like Internet2, but they are the exception rather than the rule). The MBone has collapsed, for all intents and purposes. Very few

> ISPs support multicast. As a result, wide area conferences are not really viable using multicast. However, these conferences are very suitable for LAN or enterprise conferences, where multicast is often deployed. ...".

Thus Rosenberg [61] rules out the possibility of multicasting in WANs. Moreover, MBone's guidelines, and the available resources assume that each audio session will not have more than one active speaker at a time (indeed, many of the larger MBone sessions have been effectively broadcasts rather than conferences, for instance, feeds from the NASA shuttle launches) [62, 27].

This thesis does not subscribe to any of the above views. Thus at this juncture, it is safe not to assume availability of multicasting in the Internet. The conferencing architecture should make use of multicasting whenever it is available. However, the architecture should be able to handle conferences whether multicasting is available or not.

A discussion on multicasting applications:

IP multicast has been heavily exploited in the design of a number of end-to-end protocols and multicast applications such as realtime media transport of audio and video with accompanying control information in *vat* [63], RAT [52], and MInT [64] MBone applications. Layered multicast based on layered encoding [65] and Simulcast [66] can also be used to adapt to network capacities. In the layered encoding method, media streams are encoded into a number of layers that can be incrementally combined to provide a refined version of varying quality of encoded signals. The individual layers are transmitted on separate multicast addresses. Receivers adapt to network conditions by adjusting the number of layers they subscribe to and thus enhancing perceived quality. In simulcast, the source transmits the same data stream encoded with different quality levels (in turn different bandwidth) on different multicast sessions. This scheme is called simulcast because the source transmits multiple copies of the same signal simultaneously at different rates resulting in different qualities.

A dynamic multicast tree building method is proposed in [67]. Real Time Protocol (RTP) [16] receivers can join a loosely coupled conference by using the dynamic multicast network. The architecture uses Autonomous Systems (AS) formed using what are called RTP elements in the form of a tree. RTP elements manage multicasting. Many AS in the form of a tree manage the multicast network. The delay introduced at each hop builds up when the tree height increases, thereby rendering it unsuitable for interactive services. Application Layer Multicasting [68] is similar to AS and is

used only for streaming applications and not for interactive services. Active research is being carried out in this area.

While multicasting reduces bandwidth usage by senders, in N-way conferencing, meaning N simultaneous senders, the network leading to the receiver is still faced with the problem of handling a bandwidth N times that of senders. So mechanisms are needed to reduce bandwidth at the receiver's end. A receiver may want to process only M of N streams sent, or may have a problem decoding and presenting all information. The bottleneck shifts to the receiver network and computational capacity of receivers or end terminals [51]. Multicasting is useful when there are very few simultaneous senders. This thesis takes cognisance of these conflicting requirements. By reducing simultaneous speakers for conferencing, interactivity is lost; with too many simultaneous speakers the network usage and bandwidth requirements are high. Chapter 3 performs a balancing act using intermediate servers.

2.3.5 Traffic reduction

Reduction in traffic refers to reduction in the number of audio streams in the network for a large number of simultaneous audio sources and receivers. Traffic may be reduced at the network layer by using multicast techniques. Reducing traffic at the application layer contributes to more efficient use of network resources. A few techniques other than multicasting are enumerated below:

I. Compressing the audio streams: Some compression techniques such as ITU-T G.729, G.726 and G.723 coding [69, 70] result in a very low bit rates up to 6Kbps per participant.

II. Reducing the traffic using Voice Activity Detection (VAD): There are always pauses in a speech. Dropping the silence segments results in reduction in bandwidth requirement [71] of up to 50% for a single stream.

III. Reducing the number of streams by filtering unnecessary streams: In Section 2.3.1 interactivity was discussed. If all the streams were allowed the network would be heavily loaded. A partial mixing technique is proposed in [27, 38, 29, 28] and complete mixing in [40, 41, 42]. By selecting a speaker at a time using floor control [47, 45], traffic is greatly reduced at the cost of interactivity. Therefore this thesis (Chapter 4) addresses the question of reducing the number of simultaneous streams without losing perceived interactivity of a conference using *Ethnomethodology*.

Lowering the number of simultaneous audio streams across the network is important. The cost involved in setting up, operating and distributing audio streams with simultaneous speakers increases drastically in large scale conferences compared to small scale conferences inside a subnet.

2.3.6 Quality of the conference based on packet delay, packet loss percentage and delay jitter

A crucial aspect of collaborative work is realtime service, i.e., the ability to live up to human perception regarding realism and fidelity [72]. Latency requirements expressed as total latency (delay and lag) and latency variance (jitter) critically affect the quality of audio received at the end terminals. This quality influences user satisfaction [73] in realtime collaborative works. Moreover, ergonomic studies and experience gained from the Internet demonstrate that people can use audio data as long as the information content is above some minimum level [74]. The problems of Quality of Service (QoS) for audio conversations are investigated by Maja and Lea [75]. They address networked virtual reality services over the Internet and attempt a better understanding of them in terms of QoS requirements. They present a design and implementation of a multi-user interactive application prototype and virtual audio chat, and evaluate QoS in terms of both user level QoS characteristics and communication level parameters. Such services impose some QoS requirements at the user/application level as well as on the underlying network. Anna and Sasse [76] have presented QoS measurement methodology for networked realtime multimedia systems for packet durations that are between 20ms and 80ms.

Delay of audio packets is due to transmission and queuing at the routers in the network. Variable queue sizes seen by audio packets at intermediate routers introduce delay jitter. In a packet speech system the end-to-end delay (after nullifying the jitter) is always a critical parameter in a realtime voice system. It should hence be kept well below 600ms [77] in the absence of echoes (the utility upper bound specified by ITU-T G.114 recommendation is 400ms [32, 78]) if conversation patterns are not to break down. Packets are dropped at the congested router whenever the queue overflows. Transportation error is negligible for wire-line transportation. The extent of packet loss is a primary factor determining whether a network audio stream will be intelligible to the user and therefore of any use at all. Delay and jitter play a secondary role and should also be kept under check.

The study here has considered delay and loss separately. It is interesting to note that loss and delay are some what correlated. Correlation between packet loss and delay is

reported by Moon et al., [79] for which they give an explanation as follows. As packets traverse the network, they are queued in buffers (thus adding to their end-to-end delay) and from time to time are dropped due to buffer overflow. Consider then a realtime multimedia packet stream at a buffer that is filling up fast with packets from other sources. Continuous media packets arriving at the receiver experience progressively higher end-to-end delays compared to earlier packets. When the buffer reaches its full capacity, packet losses begin to occur. The receiver of the realtime multimedia application thus sees increased delay, and eventually losses. Consider next a scenario in which packets from a realtime multimedia application arrive at a buffer that is already full. In this case, they are dropped. As other sources (e.g., TCP connections) detect congestion and decrease their transmission rate, the queue length at the buffer decreases, and packets from the realtime multimedia application begin to be queued, rather than dropped. In this scenario, the receiver sees losses followed by large, but possibly decreasing, packet delays. Experimental study done by Pointek et al., [80] using probe packets did not observe increase in Round Trip Delay just before packet loss. Some further work on this topic is reported by Haresh [81].

Hardman et al. [77, 71] have explored the relationship between packet loss and intelligibility. They argue that this is a complex relationship and that it depends on percentage of packets loss, the size of the packets and the repair strategy used to compensate for missing audio information. Permissible percentage of packet loss is dictated by the media content and the application. For instance, a foreign language is more difficult to understand than a native language. Some simple guidelines may form a useful starting point. Typical values found in literature [77, 71] suggest that audio will become unintelligible if packet loss is higher than 10% for 40ms packets with silence substitution. Upto 30% loss is tolerable [71] with a single copy of redundancy. Experimental studies on packet loss are given in [77].

There are service models to accommodate QoS requirements of realtime multimedia applications. The IETF has devised two service models for providing QoS guarantees. The first, *Intserv* [82] model accommodates QoS requirements of realtime multimedia applications. This approach is usually referred to as integrated services because it incorporates QoS requirements for realtime applications and provides performance guarantees for this class of applications at each hop for a flow. Thus scalability is poor because requirements are deployed and negotiations are operated at every node.

The second, Differentiated services, *Diffserv*, [83] enhancements to the Internet Protocol are to enable scalable service discrimination in the Internet without the need for per-flow state and signalling at every hop. A variety of services may be built from a

small, well-defined set of building blocks that are deployed in network nodes. Services may be end-to-end or intra-domain; they include those that satisfy quantitative performance requirements (e.g., peak bandwidth) and those based on relative performance (e.g., "class" differentiation) [83].

Negotiations use Resource Reservation Protocol (RSVP) [84] defined by IETF. RSVP is a signalling protocol that applications may use to request resources from the network. The network responds by explicitly admitting or rejecting RSVP requests. Certain applications that have quantifiable resource requirements express these using Intserv parameters as defined in the appropriate Intserv service specification. RSVP operates on top of IPv4 or IPv6, occupying the place of a transport protocol in the protocol stack. However, RSVP does not transport application data but is an Internet control protocol, like ICMP, IGMP, or routing protocols. Like implementations of routing and management protocols, an implementation of RSVP will typically execute in the background, and not in the data-forwarding path.

The present day Internet does not support these service models. Realtime traffic has to compete for bandwidth with other non-realtime traffic on the best effort network such as IP. In such networks there can be no assurance about the quality of service when physical limitations are present, and packet loss can be limited only through adaptation. Adaptation can be useful even for integrated networks in the case of interactive applications. Adaptation to delay jitter and loss can be found in the works of Samar [85] and Haresh [81] on Proof of Concept Setup (PoCS) platform developed as a part of this thesis.

Though this issue is of nontrivial significance in VoIP applications, in the context of conferencing it is assumed that sufficient bandwidth is provided by the network for VoIP service. This thesis does not take up the issue of handling delay and delay jitter directly. Nonetheless, it presents some insights into the dependence of packet loss and delay jitter on the packetisation interval, packet size and the load on the network, based on realtime measurements and simulations in Chapter 7.

2.4 Techniques for Handling Multiple Audio Streams

Issues and strategies for handling multiple simultaneous speakers are uncovered here.

2.4.1 Explicit floor control

Floor control or turn-taking mechanism provides a means to mediate access to shared work items. Greenberg [86] recommends that systems should "support a broad range of floor control policies" to suit the participants' needs. Floor control can be important

in many situations, such as shared screens allowing only serial interaction, or systems following strict interaction models similar to a teacher monitoring/controlling the work surface access of students. Roseman and Greenberg tackle many of these aspects [87] on GROUPKIT building. For collaborative environments several types of floor control policies such as Explicit release, Free floor, Round robin, First-In-First-Out, Preemptive, Central moderator [86] are available. However, these methods are beset with difficulties in allowing impromptu communication.

Floor control can be employed within shared workspaces to control access to shared workspace. Each system must decide the level of simultaneity to support (i.e., number of active participants at any time) and the granularity to enforce access control [44, 45, 46, 47]. In its simplest form, floor control enables floor access to only one participant at any given time and the floor is handed over when a request is incident. In the case of audio, floor control introduces a management framework around the audio session that enforces turn taking, thereby removing any potential simultaneity. Consequently[3], ambience of the application becomes suffocating or gagging for the users. This can happen even if there are more floors because, the person who wants to speak may not have a floor.

Even though explicit floor control may be suitable for some applications such as broadcast by a panel, it is inherently difficult to implement and maintain for a system of many-to-many active participants. When large scale groups are to be allowed, implementation of these techniques is cumbersome. Making a policy for floor allocation without human intervention is not simple in a large conference where all members can request and be granted the floor.

2.4.2 Push-to-talk

Collaborative system interface usually comprises a window showing a list of participants in the audio conference, gain and volume controls, and power meters to indicate speech activity, etc. In order to talk, the pointer has to be placed in the window, and the mouse button clicked (like a walky-talky). This mechanism is known as 'push-to-talk' [88]. Only after this explicit action is the participant allowed to speak to others. This is the default approach used by the current MBone based tools such as *vat* [63] and RAT [71]. These tools enable every participant to hear everybody else in the conference simultaneously. This allows users to make very careful choices of whether to speak or not, avoiding significant amounts of simultaneous speaking in many more restrained

[3]However beautiful the strategy, you should occasionally look at the results.

- Sir Winston Churchill

contexts. Nonetheless, it is liable to remove subtle non-verbal cues and sounds. Further, conversation and interaction becomes slower and unnatural due to a conscious turn-taking activity required in order to be heard. This reduces the spontaneity and interactivity of the conversations. When many participants speak simultaneously the network is flooded suddenly thereby causing disruptions due to packet loss. Silence suppression is another form of handling simultaneous speakers. However, it can be efficacious only to some extent. This thesis views this mechanism to be no more than a bandwidth saving method.

2.4.3 Audio Processing Unit (APU)

Kyeong-yeol, et al. [89], proposed an Audio Processing Unit (APU) for conferencing. It is a hardware implementation of Multipoint Control Unit (MCU) of H.323 taken up in Section 2.5.1. MCU is used for multipoint audio-video conferencing systems. This is a centralized unit, which takes a fixed number of audio sources from participants and distributes to each of them a specific mix of selected streams excluding their own to avoid echo. It therefore performs total mixing on a dynamic set of input streams. The APU selects four simultaneous speakers on the basis of individual sample energy. Once selected a participant holds the floor for a certain interval to circumvent frequent changes, thus hindering interactivity. It is implemented using a DSP chip. Its scalability is very limited. A refined selection algorithm developed in Chapter 5 of this thesis may be used in APU for better performance.

2.4.4 Comparison of methods

Ways of handling multiple simultaneous speakers were discussed above. Different techniques allow natural audio communication between participants to different degrees. More specifically, these techniques have different ways of determining as to which speakers are heard in the conference. Support for large numbers of concurrent speakers is provided only in an open microphone situation where multiple voices are transmitted as a mixed single speech signal. With floor control, push-to-talk and APU techniques not all streams are selected. Floor control and APU techniques allow full transmission of the streams but only for the selected speakers. Whereas floor control limits spontaneity, push-to-talk imposes an unnatural constrain for requesting permission.

All the techniques described above limit the number of simultaneous audio streams transmitted in the network. The number of simultaneous streams is explicitly bound in floor control and APU techniques though not in the rest. Push-to-talk and silence

suppression result in the number of streams equalling to the number of active speakers, while the open microphone technique results in the number of streams equalling to the total number of open microphones. This means that there are techniques that allow potentially very large numbers of concurrent audio streams transmitted in the network whenever applications have large numbers of users.

APU [89] is not scalable as it has a central unit and uses hardware support to decide which of the streams are to be mixed. The decisions are based on current energy of the stream. High energy noise spikes may abruptly cut off a speaker and make the mixed stream unintelligible. However, spontaneous conversations are allowed in APU and open microphone techniques. The number of simultaneous streams is set to four in APU. This thesis takes a cue from APU and improves upon it (Chapter 5) in order to build a distributed conferencing architecture. In addition, conversational analysis presented in Chapter 4 is used to set an upper bound on the number of streams sufficient for interactivity.

2.4.5 Mixing of streams

Network traffic increases as $O(n)$ for multicast enabled networks, and as $O(n^2)$ for networks without multicasting. Hence mixing streams at intermediate nodes reduces traffic. In this method, the number of simultaneous speakers can be unlimited as all streams are mixed. Thus interactivity can be provided without any hindrance. In addition, mixing can be at intermediate servers or nodes or can be done hierarchically in the conference setup. Two studies are taken up for discussion here.

2.4.5.1 Hierarchical mixing

In order to support applications such as a tele-orchestra involving a large number of simultaneously active audio streams, Rangan, Vin and Ramanathan [40, 41, 42] propose a hierarchical mixing architecture. In such applications all streams are mixed. They compare a distributed hierarchical mixing architecture with centralized mixing. They exploit the comparative durations required for transporting a packet from an end terminal to a mixer and for mixing two streams at a mixer. Depending on the ratio of these two times, they propose different architectures for media mixing.

In this mixing hierarchy participants constitute leaf nodes and the mixers, non-leaf nodes (See Figure 2.2). The mixer that is at the root of the hierarchy forwards the final mixed packet to each of the leaf nodes. This can also be done by multicasting the packet from the root, if supported, to avoid delay during hops. Alternatively, packets can be forwarded through intermediate mixers (Application Level Multicasting (ALM) in some

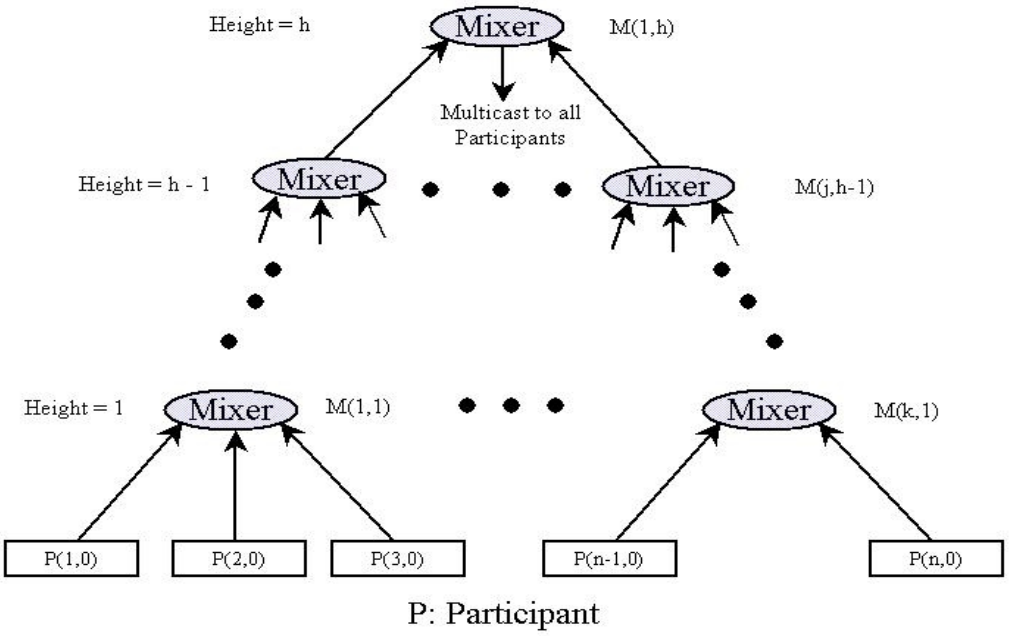


Figure 2.2: **Hierarchical mixing architecture**

respect). Like centralized mixing, this approach does not permit each listener to create a personalised mix of streams since all streams are mixed at all times. The authors show that it is an order of magnitude more scaleable than purely centralized or distributed architectures. Tree architecture is highly scalable compared to a centralized mixing and completely distributed mixing, wherein all end terminals see all others packets. The root node would also have less number of streams to mix as the intermediate mixers in the tree would have mixed streams from their children nodes.

The number of hops increases with the height of the tree in this architecture. Thus it is not well suited for interactive and collaborative applications. Each user does not have the flexibility of controlling and mixing independent audio streams. This approach is blind to underlying network conditions as it relies only on pre-configured static mixers.

2.4.5.2 Distributed Partial Mixing (DPM)

Redenkovic et al., in their recent work [27, 38, 29, 39] identify the importance of permitting speech streams from many participants. They consider situations in everyday life where a large group of people 'speak' at the same time. Audiences at performances and spectators in sports provide the most extreme examples, with up to thousands

engaging in cheering, chanting, jeering and singing together. They reason out that any relaxed social setting that involves sizeable groups of people is also likely to involve a significant number of simultaneous speakers; for example, interruptions, non-verbal speech cues [90], shouting out answers to questions, dynamically forming conversing sub-groups within a large space. In addition, they take an example of a TV game show "Out of This World" (OOTW) [37]. After studying patterns of user activity, inclusive of audio activity statistically, and analysing system logs, they conclude that overlapping audio transmissions from several participants is common. In principle they allow all streams without mixing at intermediate nodes.

The presence of many simultaneous speakers has profound implications for the bandwidth requirements of an audio service. Each speaker independently introduces a new audio stream that has to be accommodated by the network and that also has to be processed by each recipient's local computer. Whereas multicasting reduces the bandwidth required to distribute a single audio stream to many listeners, it does not deal with the many different audio streams that are produced by simultaneous speakers. From a user's point of view, an ideal multi-party audio service would present an independent audio stream from each individual speaker at a satisfactory quality with minimum latency and with no jitter. Receiving independent streams would allow each listener to create a personalised audio mix. However, this is a highly ideal situation and in this case the network cannot afford to carry full traffic of many participants as the realtime service demands more bandwidth with a tight bound on packet loss and delay. Thus to reduce the traffic on shared links from participants they propose Distributed Partial Mixing (DPM). An example is shown in Figure 2.3. As the name suggests, mixing of audio streams plays a key role. Mixing multiple streams involves digitally summing the corresponding audio samples from the incoming streams and then normalising. Mixing (say, in a server, filter or proxy) in the network actually decreases the number of audio streams. The benefits of any mixing-based approach are:

I. Mixing can drastically reduce network traffic for many simultaneous speakers. This makes it efficient with its support for efficient distribution of audio streams in the network.

II. Mixing imposes no constraints on individual speakers unlike "gagging" as in floor control.

III. The system's view of "speaking" is the same as that of the user; naturally, mixing is more effective for supporting the most natural audio communica-

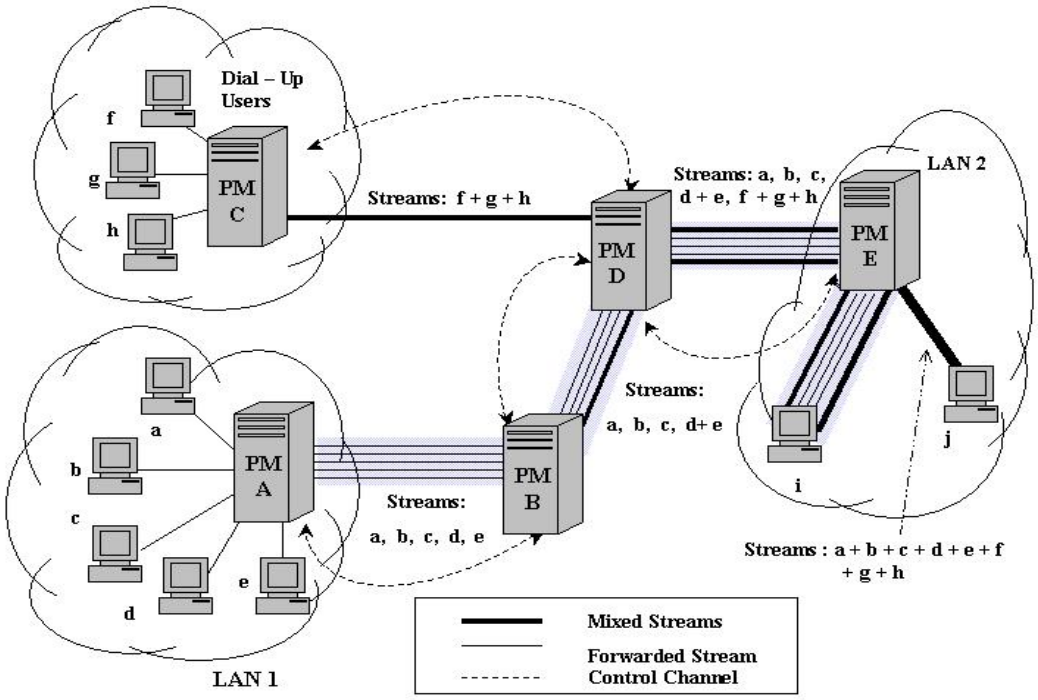


Figure 2.3: **Distributed Partial Mixing Architecture**

tion.

Although mixing all streams reduces the traffic, it would result in loss of spatialism by disallowing customised mix of streams at the receiver with user-specified weights. For this they propose partial mixing of streams (Figure 2.3). Consequently two questions that arise out of DPM design are: (1) How is an optimal mixing achieved? i.e. how to keep the possible number of independent streams a maximum while having an efficient aggregation of audio streams? (2) How can the audio streams be mixed in a way that is adaptive and friendly towards other traffic in the network, yet minimising packet loss? They claim that distributed partial mixing can be performed by a number of Partial Mixing (PM) components that might already exist in the network. PM components may exist as dedicated servers or as additional roles played by other end terminals. End terminals typically communicate via these PM components, sending their audio to, and receiving other participants' audio from, a nearby partial mixer. Partial mixers in turn form a fully connected network (e.g. a tree) to achieve end-to-end distribution of audio streams. PM extends the traditional concept of mixing to render it more dynamic and flexible. It is proposed in [27, 38, 39] that unlike total mixing where the entire set of received streams is mixed into a single output stream, partial mixing dynamically chooses only a subset of the available audio streams to mix at any given time instant

and forwards it with other unmixed streams. Selection of number of streams for mixing is based on network conditions. Thus instead of producing a single output stream in all cases partial mixing produces varying numbers of streams in different situations. A rate adaptive method with TCP fairness is also proposed in [27]. Despite these features, there are some limitations of partial mixing and mixing in general, and is taken up next.

2.4.6 Limitations of mixing

Mixing of audio streams digitally involves summing up all streams at each sample instant with some weight for each stream. The weights sum up to 1 to avoid signal saturation. A discussion on digital summation is given in Section 4.6.1. Mixing has the following problems [27, 39]:

1. The mixed signal has a lower SNR, and a lower dynamic range than individual streams.

2. Fidelity is poor since it includes noise from all the separate streams in fewer bits.

3. Mixing may not work well with some ultra-low bandwidth codecs due to their compression algorithms.

4. Mixing in stages introduces additional delay as audio streams have to be processed by the mixers.

5. Spatialisation and other aspects of control of individual listener over each audio stream are lost.

6. While mixing digitally, the volume level of individual streams is reduced to avoid saturation.

7. Mixing is irreversible and the individual streams cannot be recovered at a later stage.

8. Mixing introduces more components (servers) and increases the complexity in the network with corresponding hardware and realtime control requirements.

In hierarchical mixing the total number of hops depends on the height of the tree. Delay increases with the height of the tree; all the above drawbacks of mixing come into

picture. DPM is an adaptive method that reduces total number of streams mixed at intermediate nodes and thus tries to preserve spatialism. Spatialism is preserved only when there are a few participants or with less number of simultaneous speakers since with many speakers, mixing eventually leads to loss of spatialism. There are no fixed rules for selecting the number of streams for mixing. It is difficult to judge the streams on the basis of criteria given in [39]. In addition the weight assignment while mixing cannot be kept constant if the number of streams varies depending on the network load or the number of speakers. Customised mixing at the end terminals is difficult when many streams from varying number of sources are to be mixed since the user will have to manage those streams for setting the weights for mixing most of the time.

2.5 Standards

There are two major standards that govern VoIP services. They are briefly discussed here to evaluate recommendations for audio conferencing service.

2.5.1 ITU-T H.323 standard

H.323 is a traditional circuit-switched approach to signalling based on the ISDN Q.931 protocol and earlier H-series recommendations such as H.245 for control, H.225.0 for connection establishment and H.332 for conferences. H.323 [91] recommendation is comprehensive and flexible, and can be applied to voice-only handsets as also for full multimedia video-conferencing stations. It defines multimedia communication standards for the IP-based networks. Designed to compensate for the effect of highly variable LAN latency, it allows customers to use multimedia applications without changing their network infrastructure. By providing device-to-device, application-to-application, and vendor-to-vendor interoperability, H.323 allows customer products to inter-operate with other H.323 compliant products. Here only the conference parts of the H.323 standard specifications are considered briefly. The H.323 audio conferencing architecture, shown in Figure 2.4, defines Multipoint Control Unit (MCU) and Participants (also called as clients/end terminals), which are the key elements in the architecture. The MCU is an endpoint in the network, which provides the capability for three or more participants and Gateways to participate in a multipoint conference. It may also connect two participants in a point-to-point conference, which may later develop into a multipoint conference. The MCU consists of a mandatory Multipoint Controller (MC), and optional Multipoint Processor (MP). In a decentralized conference, the participants talk to each other using multicast or unicast capability of the network. The selection of

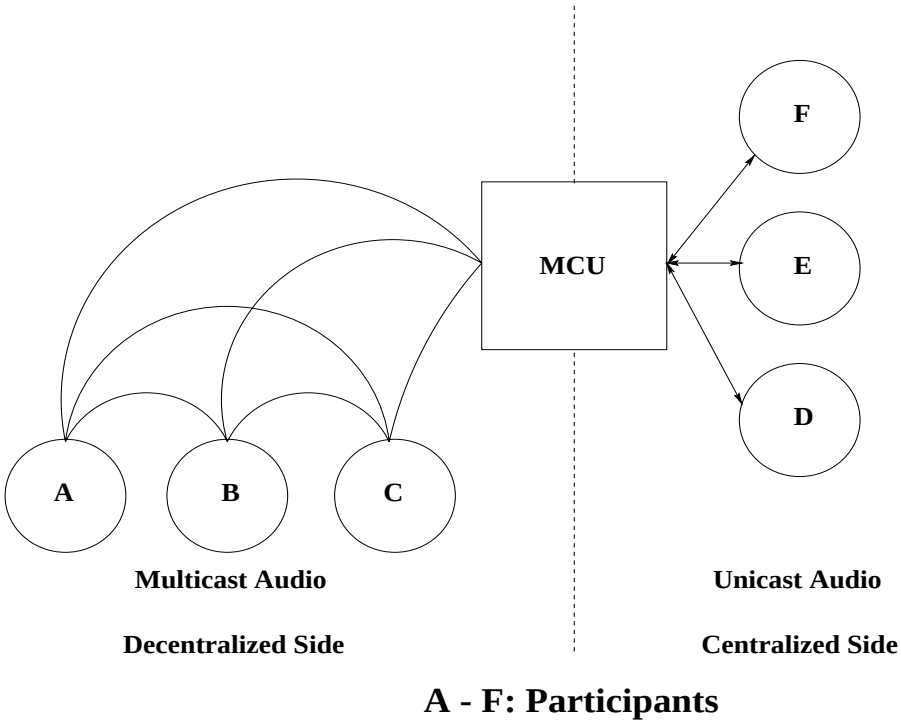
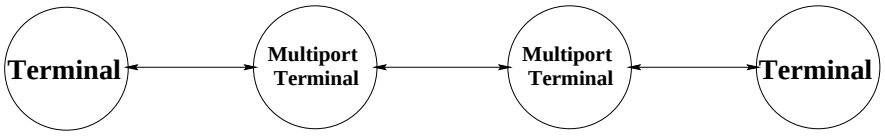


Figure 2.4: **Decentralized and centralized conference (from H.323)**

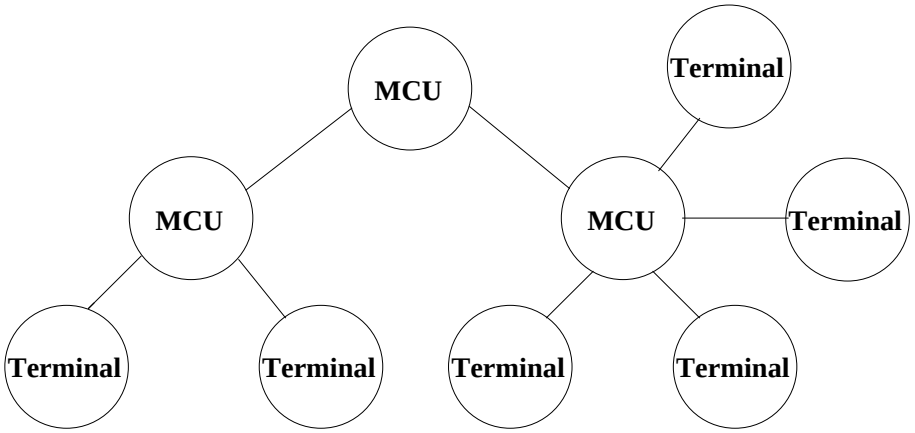
packets and mixing is done at each participant. MC does not deal directly with any of the media streams. In centralized conferencing the MP mixes, switches, and processes audio, video, and data packets under the control of the MC. The MP may process a single media stream or multiple media streams depending on the type of conference supported.

2.5.1.1 Multipoint Controller (MC)

The MC performs H.245 [92] multipoint control functions to support conferences between three or more endpoints in a multipoint conference. It carries out the signalling between Multipoint Processor and endpoints in a multipoint conference. An MC sends a capability set to endpoints in the conference indicating the operating modes in which they may transmit. It may also revise the capability set that it sends to the participants as a result of participants joining or leaving the conference, or for other reasons. The choice of conference mode may be limited by the capability of the endpoints or the MC.



(a) Chain Connection with Terminals acting as Data Bridges



(b) Multipoint Topology

Figure 2.5: **Examples of multipoint conference configurations showing various connection topologies and mode types (from T.120)**

2.5.1.2 Multipoint Processor (MP)

The MP receives audio, video and/or data streams from the endpoints involved in a centralized or hybrid multipoint conference. The MP processes these media streams and returns them to the endpoints. It may process one or more media stream types. An MP that processes audio prepares N audio outputs from M audio input streams by selecting, mixing, or a combination of these (Figure 2.4). Audio mixing requires decoding the input audio to linear signals (PCM or analog), performing a linear combination of the signals and re-encoding the result to the appropriate audio format. The MP may eliminate or attenuate some of the input streams in order to reduce noise and other unwanted signals. Each audio output may have a different mix of input signals providing for private conversations. End terminals of participants assume that their own audio packets are not present in the audio stream returned to them by an MP. Mixing in a participant makes it possible to suppress the participant's own audio stream avoiding the echo. H.323 does not address the echo problem and assumes that the mixed stream is without any echo. Communication between MC and MP is not standardised. N audio streams may be mixed at the MP if there are some dumb participants in the conference.

2.5.1.3 Limitations of H.323

H.323 does not address the scalability issue of a conference. The architecture proposed a cascaded and daisy chain topology [93] (Figure 2.5). It can be shown that this architecture cannot scale to a large conference. The new version of the standard is for essentially a tightly coupled conference with a large number of participants listening as passive participants [53] (like lightweight-sessions-style conference). Limited extensibility is through the "nonstandardParam" field allowing different vendors to develop their own extensions. The problems that affect the conferencing directly or indirectly are listed here [94, 95, 96].

a) Not many places have "nonstandardParam" field.

b) Cannot add a new value to an existing parameter.

c) There is no mechanism for terminals to exchange information about the extensions they support. H.323 requires full backward compatibility from each version to the next [95]. This is an obstacle for developing new features.

d) WAN issues such as wide area addressing and user location are not considered.

e) H.323's complexity also stems from its use of several protocol components. There is no clear separation of these components. Many services require interactions between several of them such as H.245 and H.323 for conferencing along with ITU-T T.124 [97] for generic conference control.

f) Inter-working with other protocols is difficult.

g) For large numbers of domains and complex location operations, H.323 has scalability problems. It provides no easy way to perform loop detection in complex multi-domain searches (it may be done statefully by storing messages, but this is not scalable [95]).

h) The latest version of H.323 has defined the concept of cascaded MCs, allowing a very limited application layer multicast distribution tree of control messaging. This improves scaling to an extent, but for even larger conferences, the H.332 protocol defines additional procedures. This means that three distinct mechanisms exist to support conferences of different sizes.

The Internet community feels that the H.323 standards for large-scale conferencing are complex and *not easily scalable* [95].

2.5.2 Session Initiation Protocol (SIP)

The SIP standard defined by RFC 3261 [98] of IETF is a major step for implementing VoIP in the Internet along with RTP [16] for media streams (even H.323 uses RTP). It is a text-based and HTTP-like signalling protocol. It identifies the destination via a SIP URI of the form *sip:vprasad@mydomain.com* while H.323 uses alias addresses.

SIP is an application layer control protocol that can establish, modify, and terminate multimedia sessions such as Internet telephony calls. It can also invite participants to already existing sessions such as multicast conferences. Media can be added to and removed from an existing session. SIP transparently supports name mapping and redirection services which supports personal mobility; i.e., users can maintain a single externally visible identifier regardless of their network location using Home Domain. SIP uses three major entities: (i) *Registrar*: A server that accepts REGISTER requests and places the information it receives in those requests into the location service for the domain it handles. (ii) *Proxy Server*: Proxy Server is an intermediate entity that acts both as a server and a client for requesting on behalf of other clients. Requests are serviced either internally or by passing them on, possibly after translation, to other servers. A Proxy interprets and, if necessary, rewrites a request message before forwarding it. (iii) *User Agent(UA)*: A logical entity that can act both as a user agent client and user agent server (that hosts the front end for the users/participants).

SIP does not offer conference control services such as floor control and does not prescribe how a conference is to be managed. The concept of Conference Server is not included as UAs are capable of doing the work similar to that of an MP in H.323. A Conference Server can be modelled as a UA. SIP can be used to initiate a session that uses some other conference control protocol. Since SIP messages and the sessions they establish can pass through entirely different networks, SIP cannot and does not provide any network resource reservation options. However, there is a proposal and a framework laid down in [99] for a centralized SIP conference. Kundan Singh [100] et al., propose some conference models in SIP using servers called SipConf entities. Some SIP Conference models are considered in the following section.

Conference models using SIP

The core SIP specification supports many models for conferencing [100] and an in-depth discussion is given in a recent IETF draft [61]. In server-based models, media streams are mixed by a server. The initiator of the conference uses Session Description Protocol (SDP) for defining media capabilities and other details about the conference. In a server-less conference, mixing is done at the clients and most often each client is

Table 2.1: **Comparison of SIP conference models**

Conference Type	Signaling	Media	Scalability
End System Mixing	Tree	Tree	Small
Multicast	Peer-Peer	Multicast	Large
Dial In Dial Out	Star	Star	Medium
Hybrid	Star	Mesh	Medium

capable of mixing the media streams.

First, about the server-less models: In *End-System Mixing*, only one end terminal (or UA) handles the signalling and media mixing for all others. This is clearly not scalable and causes problems when that particular client wishes to leave the conference. In the *Users Joining* model, a UA which adds a new party acts as a mixer to the newly added UA. Thus the architecture grows as a tree, since each invited party constitutes a new branch in the distribution path. This leads to an increasing number of hops for the two remote leaves at the ends and hence scalability is compromised. Another option would be to use multicast for conferencing but multicast is not enabled over Internet; it is possible only on a LAN presently.

Second, the server-based models: *Dial-In Conferences* have UAs connecting to a central server which handles all the mixing. This model is not scalable as it is limited by the processing power of the server and bandwidth of the network. *Ad-hoc Centralized Conferences* and *Dial-Out Conference Servers* have similar mechanisms and problems. *Hybrid models* involving centralized signalling and distributed media, the latter using Unicast or Multicast, raises scalability problems as before. However, an advantage of this scheme is that it can use any proprietary conference control. In a *large scale multicast conference* a UA would be given information about a conference after fixing the parameters for the conference using SDP and may be announced using Session Announcement Protocol. The multicast addresses can be dynamically obtained and a conference can be started anytime. Participants are loosely coupled; their relationship is always point-to-point. Each UA sends its packets to the group address and receives packets from other UAs on the same group address. In this model, a new user can be

invited using INVITE along with the SDP or a user may join using SDP. The limitations are: (i) Periodicity of RTCP reports; (ii) Security and lack of multicast capabilities of the existing network; and (iii) End terminals and their networks being potential bottlenecks. Though this method is scalable Rosenberg et al. [61] are skeptical about the deployment of multicasting-enabled routers in large scale (see Section 2.3.4). Table 2.1 compares the different methods discussed. Thus SIP at present is not directly capable of handling all the issues listed in Section 2.3 for conferencing. However, SIP is the most adaptable standard that can handle small and large conferences in a flexible manner allowing application developers to have their own way of handling and maintaining a conference.

The architecture proposed in this thesis is not entirely different from the above standards. However, attempt here is to incorporate flexibility with respect to the issues listed in Section 2.3. In the next section some major implementations are considered very briefly.

2.6 Implementations

There are many pilot implementations and some commercial applications for VoIP conferences.

2.6.1 Microsoft NetMeeting

Microsoft NetMeeting [101] is a real time desktop conferencing and collaboration system that offers multipoint data conferencing, text chat, whiteboard, and file transfer, as well as point-to-point audio and video. It is a standard part of the latest Windows operating system. Adaptive bandwidth management is used to ensure that the best possible use of resources is made at all times. NetMeeting captures video frames at a resolution of either 160×120 or 320×240. For audio, NetMeeting offers bit rates between 44.8Kbps and 64Kbps and defaults to a lower value in this range. H.263 and G.723 standards are used. Audio and video quality are affected by a number of issues including processor power and available bandwidth. NetMeeting offers a highly configurable choice of audio and video qualities at low and medium quality range. Solutions are dependent upon the situation they are being put to use.

By default NetMeeting supports direct communication between two remote participants. NetMeeting can potentially operate with any H.323 conferencing product or service over TCP/IP connections. Audio is sent using RTP over UDP/IP. These products and services can be separated into the following categories:

1. Audio and video conferencing clients

NetMeeting is capable of connecting to any audio or video conferencing client that supports the H.323 specification. Depending on the availability of compatible audio and video codecs, H.323 products can potentially interoperate and exchange audio and video information. For example, NetMeeting is fully interoperable with the Intel® Internet Video Phone providing high-quality real-life video and audio conferencing from a local desktop computer.

2. Conferencing servers or bridges

Conference servers or bridges can host multipoint audio and video conferencing for H.323-compatible products. Some servers (MCU) and bridges may also host audio and video conferencing in conjunction with data conferencing such as program sharing and file transfer. Service providers can offer an advanced network of audio, video, and data conferencing bridges, professional conference administrators, and special features such as password security. NetMeeting is an example of H.323 standard implementation. As servers are mostly centralized the conference is not scalable.

2.6.2 Distributed Interactive Virtual Environment (DIVE)

DIVE [102, 103] is an Internet based multi-user Virtual Reality (VR) system where participants navigate in 3D space and see, meet and interact with other users and applications. It is an experimental platform for the development of virtual environments, user interfaces and applications based on shared 3D synthetic environments. DIVE is especially tuned to multi-user applications, where several networked participants interact over an Internet.

DIVE offers a highly functional collaborative environment which can be tailored to provide support for whatever activities are required to take place in it. By default the environment supports spatially oriented audio communication between participants. It is possible to discern the location of the environment from where a sound is originating. An "avatar" (a graphical object, a 3D icon) embodies each person and viewpoints. The use of textures makes it possible to construct detailed and realistic environments and using scalability techniques very large models can be loaded into DIVE. For example, there is a 160 square kilometre model of London [104]. DIVE also supports realtime streaming of video images. The quality of the audio and video supported is not the same as the dedicated videoconferencing systems, with audio encoding ranging from μ-Law to GSM, and video images are small with low refresh rates.

A user or application is represented by an *Avatar* [105] which can move in a 3D space. The icon is associated with a space around it called the "aura" (Section 4.2.2 has some more details). Aura is used to establish communication between agents (users or applications). Communication is established between two parties when the auras of their avatars intersect and if they are in the same *world* (that are disjoint geometric spaces). Communicating with applications essentially means enabling them. For users it means perceiving other users by talking to and seeing them. Therefore there is a form of floor control based on synthetic visual space intersections. All audio streams of users in an aura are mixed. The network connection between the entities relies either on a site being connected to the MBone, or else a dedicated service developed called the DIVEBone (similar to MBone). This allows multicast tunnels to be created between multicast-unaware sites using a central server and connecting proxy servers (often referred to as Application Level Multicasting - ALM). Greenhalgh discusses in detail on the network requirements of DIVE like systems in [106].

As DIVE is constructed using multicast techniques, in principle, it scales very well to support a large number of simultaneous users. In the case of large number of participants inside a single aura the number of simultaneous speakers increases without any bound, thereby bringing down the quality of conference as well as increasing the network traffic due to many simultaneous speakers. In DIVE system all streams are mixed.

2.6.3 *vat*, RAT and MINT

2.6.3.1 *vat* - Visual Audio Tool

Visual Audio Tool *vat* [63] is an audio conferencing application developed by Van Jacobson of the Network Research Group of Lawrence Berkeley National Laboratory. It is one of the earliest MBone tools for collaborative work. *vat*, is a realtime multiparty multimedia application for audio conferencing over the Internet. It is based on RTP developed by the IETF's Audio/Video Transport working group. RTP is an application-level protocol implemented entirely within *vat*, and requires no special system enhancements to run it. Although *vat* can be run point-to-point using standard unicast IP addresses, it is primarily intended as a multiparty conferencing application. To make use of the conferencing capabilities it must support IP multicast, and ideally, user network should be connected to the IP Multicast Backbone (MBone).

vat is a lightweight session tool that uses Application Layer Framing (ALF) proposed by Dave Clark [107]. It was proposed to use multicasting for everything except

a point-to-point call. For reliability some repair mechanisms are also provided. The protocols used are application specific. The design was around simplex communications and distributed consensus. For example, a receiver may ignore a sender. The design philosophy of *vat* tool was to compose the complete system with many modules – Floor Control, Communication Block (like RTP alone handles all the audio packet transportation and communications), Conference Session Creation, etc.

By and large, *vat* mooted centralized and uncoupled or loosely coupled floor control. They propose [108] two models, viz, 'sender control' model and 'receiver advise' model. Sender control breaks the speech of the speaker badly but receiver advice is smooth. Since MBone is still an 'overlay' on the Internet it is not trivial to get hooked up to MBone and requires cooperation from local and regional network.

2.6.3.2 Robust Audio Tool

RAT is an open-source MBone audio conferencing and streaming application developed by the Multimedia group at UCL [109]. It requires no special features for point-to-point communication, except a network connection and a sound card. RAT allows users to achieve realtime multi-way communication over the Internet. It was initially intended for use in multi way conferences, but more recently is used as an Internet audio broadcast application, by radio stations in the US and elsewhere. RAT can also be used in a point-to-point manner, and as a transcoder between networks of differing capabilities; for e.g., for mobile access to the Internet. It is one of several tools that have been developed within the research community.

The emphasis of work in RAT has been on maximising audio quality despite inherent problems of packet transport, processor scheduling and audio capabilities of the end terminal. It also serves as a platform for audio research. For maximum flexibility, RAT is designed with modular approach and many of the underlying subsystems have alternative complexity/capability implementations. RAT can also leverage off platform-specific features, such as software and hardware audio codecs, various audio devices within a platform, and alternative protocol stacks.

The important features of RAT, compared to other Internet audio tools, are that it is capable of supporting multi-rate processing, has no restrictions on audio frame duration, and supports multi-channel audio of both fixed and variable size audio frames. RAT features a range of different rate and quality codecs, receiver based loss concealment to mask packet losses, and sender based channel coding in the form of redundant audio transmission. Supported audio quality range from 5.6Kbps using Linear Predictive Coding (LPC) to PCM at 64Kbps with many options in between.

For multiparty conferencing RAT uses IP multicast and therefore all participants must reside on a multicast capable network. RAT is based on IETF standards, using RTP above UDP/IP as its transport protocol, and conforming to the RTP profile for audio and videoconference with minimal control. More discussions on different implementations are provided by Adrian Bullock [110].

2.6.3.3 Multimedia Internet Terminal

Multimedia Internet Terminal (MInT) [64] is a flexible multimedia tool set that allows establishment and control of multimedia sessions across the Internet. The system architecture is fully distributed with no central components. It has audio transmission and reception for qualities ranging from GSM to CD quality based on the NEVOT and *vat* tools. Video transmission and reception is based on Video Conferencing Tool - *vic* [111]. Complexities of the individual tools are hidden by an integrated conference control instance. It also has floor control, voting based on MPOLL, RSVP reservation capabilities for all applications. Invitation of remote users is based on SIP.

MInT identifies problems with the two existing approaches. First, large, monolithic tools that support different tasks such as video, audio and application sharing, consist either of a single program or a tightly integrated set of applications. They can only interoperate within the set. It is difficult to add new features to such tools or upgrade and replace the media agents. Secondly, in loosely coupled tools, such as the Internet MBone tools, a distinct media agent typically handles each medium. An external conference controller is needed for floor control, joining and leaving a conference and starting appropriate agents. With such an approach, the media agents can be easily replaced, updated and reused. The main drawback of this approach is that once the conference controller has started a media agent, the media agent is on its own and the controller is no longer in control of it. This means that the user has to employ a different user interface for each application with all of the interfaces doing in part common tasks such as initiating or terminating a service or displaying members of a session. A more serious drawback of the loosely coupled tools approach is the lack of interaction between the different tools. MInT uses a new approach that combines monolithic and the loosely coupled tools. Using a simple communication protocol named Pattern Matching Multicast (PMM) [112] various media agents can communicate with each other without depending on each other. Communication between different media agents is established using local multicast (multicast using $TTL = 0$).

The Integrated Session Controller (ISC) is the central control entity in MInT and it provides the user with the necessary interface for initiating and terminating a session.

Furthermore, it enables the user to control the media agents. With PMM protocol a session controller (ISC) can start, terminate and control different agents. Use of this approach eases the integration of different media agents, and this can be very helpful for various reasons like lip synchronisation or audio-video interaction. It also has a Reservation agent (which uses RSVP), a Floor control agent and a Voting agent.

With respect to the media the MBone applications continue to exist as they are. The limitations of using MBone applications from the viewpoint of multicasting availability and requirement of a central controller for allocating the floor continue to remain.

2.6.3.4 Discussion on implementations

Many of the available implementations were presented above. Some tools use standard protocols and others do not. Even *vat* and RAT are not integrated with protocols such as SIP or H.323. They are mainly audio modules by themselves and can be used with any standard. Floor control and multiple speaker problems are not addressed in any of these implementations. Whereas *vat* proposes some loosely coupled control, DIVE allows all the streams inside an aura.

Transportation of audio to multiple destinations, multicast support was always assumed. When there is no local support for multicast it is possible to tunnel multicast packets over a unicast UDP channel from a site with MBone access to one without this access. Most current commercial Internet audio streaming applications use application-level "reflectors" to forward packets from source to receivers, or to other reflectors. Cascades of reflectors are used for broadcasting to large audiences, but since reflectors are manually placed on amenable hosts rather than in network routers, the distribution tree may be sub-optimal. Reflector schemes are bandwidth inefficient, and they neither scale well nor are they robust to network outages. Since many Internet Service Providers (ISPs) do not support multicast, there is currently no alternative to reflector scheme. Varakliotis et al. [113], wish (in 1998) that multi-way multimedia delivery over the Internet will be via multicast. However, that is nowhere in sight even in the year 2003 [61].

2.7 Conclusions

The issues that are important for a voice-only conference were discussed in this chapter. Various techniques, standards, implementations were analysed in depth. Table 2.2 summarises these techniques for the issues considered in this chapter. Many methods considered in Table 2.2 are not strictly comparable as they are based on different

concepts.

Techniques/implementations/standards listed in the table identify many of the requirements for conducting a conference. Mixing architectures are highly scalable. They support highly interactive conference as all streams are selected. In DPM technique mixing is chosen as a method to regulate total traffic, thus making it TCP-friendly. The drawbacks of mixing are the loss of spatialism and after mixing the streams cannot be recovered. This leads to unpleasant situations wherein a stream that does not have valuable speech content would also get mixed. Consequently, it results in lowering the volume of the mixed speech and if not mixed increases the bandwidth as in the case of DPM.

H.323 is a useful standard that defines all the necessary requirements for conducting a conference (video and/or audio). But scalability is poor. Interactivity is also poor because of floor control mechanisms. Moreover, though a large number of H.323 deployments are seen, presently Internet community is moving towards decentralized loosely coupled solution using SIP.

It seems that SIP is the standard widely accepted at present. There is no explicitly defined architecture for conferencing. It allows a User Agent (UA in SIP terminology for a client) to act as a conference server. SIP allows a loosely defined conference setup and allows implementers to have different conferencing models for deployment. Conference paradigms of *vat*/RAT/MInT blend well with SIP philosophy. The speaker selection is primarily using some proprietary floor control. Thus for large conferences interactivity is poor.

DIVE is from a different school of thought. 3D virtual worlds (that are disjoint geometric spaces) are stored and displayed on users' screens. Participants are in different parts of a 3D world and they are represented by *Avatars*. Participants are able to interact with only those who are near to them, i.e., within their aura or world. It is a user-friendly tool with enhancement in interactivity due to visual Avatars. It allows for Application Level Multicasting (ALM) when the router does not support multicasting [114]. It cannot scale up for large number of participants since visual effect (stored visual space) is required to aid interactions. Nonetheless it allows fully interactive environment for those inside a world. A study by Eckehard [24] shows that the improvements in interactivity by adding video is very minimal. Thus for a large conference it is advisable to opt for a voice-only conference first. Since DIVE requires visual aid for interaction it is not easy to deploy and service a large customer base for voice-only conferencing.

Table 2.2: **Comparison of Techniques, standards and implementations for audio conferencing**

Techniques/ Standards/ Implementations	Issues						
	Speaker Selection[a]	Interactivity	Customized Mixing	Scalability[b]	Multicast Requirement	Traffic Reduction[c]	Handling Delay & Jitter[d]
Hierarchical Mixing[e]	All Select	High	Not Possible	High	Not Mandatory	Through Mixing	May be handled at intermediate Mixers
DPM[f]	All Select	High	To Some Extent	High	Not Mandatory	Through Mixing	May be handled at intermediate PMs
H.323[g]	Through Floor Control	Poor	Using MPs	Poor	Not Mandatory	Not Possible	May be handled at intermediate MPs
SIP with RTP for Media[h]	Third Party Control	Poor	Note A	High	Mandatory	Not Possible	End Terminals must handle
DIVE	All in an Aura	Only within Worlds/Hierarchies[i]	Possible	Note B ALM[114]	Not Mandatory	Using smaller World sizes	End Terminals/DIVE proxy must handle
vat, RAT & MInT	Through Floor Control	Poor	Note A	High	Mandatory	Limiting no. of Floors	End Terminals must handle
APU	Based on instant Energy	Less[j]	Note C	Poor	Not Mandatory	Note D	Note D

Notes:

A: Not Applicable, since only one speaker is selected through Floor Control.

B: ALM - Application Level Multicast increase scalability. **C**: Central decision entity may provide customized mixing.

D: Not Applicable, since the Central entity is the bottleneck.

[a] Actually, Speaker Selection & Interactivity goes hand in hand; it was not identified as an *issue* earlier; it is listed seperately *here* to enable a comparision.

[b] Fully multicast network is taken as highly scalable.

[c] Any technique for reducing traffic other than multicasting.

[d] Facilitating adaptive playout buffers, reserving bandwidth, transcoding, etc., may improve the quality.

[e] [115] does not discuss conference setup.

[f] Doesn't use any standard protocol for handling the conference setup.

[g] Microsoft Netmeeting uses H.323. Hence not considered here.

[h] Loosely coupled multicast model is considered here (see Table 2.1).

[i] Users are grouped into smaller virtual 3D spatial worlds. Interactivity is within the world.

[j] Although adequate floors are available, interactivity is lost due to floor exchanges only at some periodic instants.

The APU is a centralized solution that can select N simultaneous speakers to speak concurrently. It can provide customised mix of the streams with additional hardware or computational power. Scalability is poor because of centralized approach. Interactivity is limited because speakers are selected based on energy at regular time instants and not continuously monitored. It is developed as an H.323 entity.

The requirements for a large audio conference are scalability, adequate floors for interactivity, and dynamic selection of speakers, i.e., dynamic allotment of floors. Multicast support in Internet is not available currently. Therefore there is a need for a distributed APU type system that supports dynamic speaker selection with low complexity. An appropriate number, N, of simultaneous speakers are to be supported to preserve interactivity without the help of visual systems as in DIVE. It should allow ALM, if required, for connecting to non-multicast supported domains. Since teleconferencing is one of the services of VoIP, there is a need for this service to conform to a standard for call/conference management.

The next chapter starts out with these requirements. An elegant architecture is proposed for a reasonably scalable and sufficiently interactive (more than one simultaneous speech stream with dynamic speaker selection) conference support. The focus of the efforts in this thesis is directed towards seeking solutions to the teleconferencing problem on Internet in such a setting.

Chapter 3

Architecture for Audio Conferencing on the Internet

3.1 Introduction

In the previous chapter a discussion on the existing solutions and standards for audio conferencing was presented. Also, discussions on these standards and techniques *vis-à-vis* some important issues were presented. The discussions in Chapter 2 show that there can be improvements in the way a conference service is supported on the IP network. The criteria that are used for comparison of various solutions available (in Table 2.2) are recalled below:

1. Interactivity
2. Speaker Selection
3. Customised Mixing
4. Scalability
5. Availability of Multicasting
6. Traffic Reduction
7. Handling delay and delay jitter

Some of these issues are interrelated. Trying to improve one may counter the advantage gained by the other. Two examples of such contrasting issues are given below.

Interactivity needs speakers to speak simultaneously and interactivity increases with the number of simultaneous speakers. Streams from these simultaneous speakers are

to be mixed at each client[1] for listening. Customised mixing with different weights for each stream increases perceived quality of the mixed signal. This means that audio streams from all the speakers have to be available at each client. Therefore, a number of simultaneous speakers imply designing the conferencing solution to handle setting of different weights for all these streams at the clients. However, having a large number of simultaneous speakers countermand the scalability of the conference, while reducing the number of simultaneous speakers reduces interactivity. Reducing the number of simultaneous streams reduces total traffic generated by the conference solution but at the cost of interactivity.

Partial mixing proposed by Redenkovic [27] using DPM can be a good starting point for reducing the traffic and increasing the scalability. Redenkovic proposes to keep the number of un-mixed simultaneous speech as high as possible based on the network capacity. However, it reduces the quality of voice since the number of streams mixed at any instant depends also on the network occupancy and thus varies. This causes variations in the volume level of mixed speech. Furthermore, spatial aspect of speech is lost due to mixing.

3.2 Motivation for a New Architecture

To achieve high interactivity, Ramanathan, et al., [115, 40, 41, 42] propose to mix streams from all participants in hierarchies of mixers. Redenkovic [27, 38, 29] also proposes to select all speakers who are speaking at any instant to achieve high interactivity. Redenkovic slightly deviates from Ramanathan, that is, mixing is only partial at intermediate nodes to reduce the bandwidth requirement when traffic in the network rises. With these architectures customised mixing of streams at a client cannot be supported.

Speakers are selected using a *Floor Control* technique. Floor control is a major consideration while designing a conferencing solution [44, 45, 46]. However, with floor control, interactivity is lost due to explicit instruction to the participants as to when to speak. Customised mixing is also not provided in this scenario.

A tightly coupled conference control protocol in Internet is a part of the ITU-T H.323 family [91] and is useful for small conferences. A recent IETF draft by Rosenberg and Schulzrinne [61] discusses conferencing models with SIP [98]. Implementation for centralized SIP conferencing is reported in [100].

[1]Henceforth, the term "client" is used interchangeably with the term "participant" and is nothing but the terminal of a participant.

While mixing architectures scale well, centralized mixing architectures scale poorly. With explicit floor control, scalability can be improved since simultaneous speakers are none or very limited and static. However, interactivity is lost as discussed above.

Availability of multicasting is mandatory for many MBone implementations such as *vat*/RAT/MInT. Without multicasting, transporting many streams would be difficult, thus compromising scalability. Traffic can be reduced using mixing or floor control (allowing only one stream) apart from multicasting, however, with added disadvantages described above.

Handling delay and jitter at a client is difficult with multiple speakers. RTP [16] and RTP control protocol (RTCP) [16, 116] allow some mechanisms for handling these. Nevertheless, with many streams, it is complex to handle these at the client. The solutions suggested by various researchers above with some additional discussions and remarks on them are given below.

A. ***Audio Mixing***:

In an audio conference streams from all the participants need not be mixed. Mixing many arbitrary number of streams from participants degrades the perceived quality of the conference due to reduction in the volume (spatial aspect of speech). The number of streams mixed varies with time depending on how many are speaking concurrently. This variation could lead to fluctuations in the volume of the mixed speech streams, causing severe degradation in quality. Customised mix of streams is not possible when there are many active participants. There is an upper bound (refer to Section 4.6.3) on the number of simultaneous speakers beyond which, increasing the number of speakers does not improve the conference quality. Thus it is advisable to keep the number of speakers constant. Partial mixing has problems similar to that of mixing all streams [115, 41]. Moreover, to allow tailored mix at clients, in partial mixing scheme, mixing is not done when network can afford high bandwidth for sending/receiving all the streams [27, 38]. However allowing all streams is unnecessary and undesirable even if the network can afford to support the high bandwidth requirements.

B. ***Number of Mixers***:

Intermediate mixers (DPMs) [27, 38, 29, 39], Conference Servers [99] or mixers in hierarchical stages [115, 40, 41, 42] introduce increased delay by increasing the number of hops due to stages of mixers. Interactivity is achieved at the cost of scalability.

C. ***Floor Control for an audio conference***:
Floor Control with explicit instructions for participants - commanding as to when to speak - results in conference becoming excessively a one-way speech rather than a close approximation of live, free-to-interrupt real-life conference. Conference becomes markedly artificial and one of degraded quality. Schulzrinne et al. [16], assume only one participant is speaking at a time. In this case, if applications are implemented with some floor control, the service becomes 'gagging' for the users.

D. ***Centralized servers and completely distributed multicast approach***:
The centralized approach cannot support large conferences since the load on the centralized server increases linearly with the number of clients. Moreover, the network connection leading to the server has to afford a high bandwidth, which is proportional to the number of clients to be supported.

With completely distributed design even with multicasting, clients will have to parse many streams and also the traffic on clients' network increases. Even for the sake of argument with a large scale multicast conference, it is very artificial to assume that participants behave very well and only one speaks at a time without any floor control. Furthermore, there is a need to allow impromptu speech.

With the above remarks, it is seen that there is a need to propose a new architecture – that overcomes all the limitations listed above – accommodating conflicting criteria listed in Section 3.1 to the extent possible.

3.3 Problem Statement

The architecture for VoIP conferencing should provide *sufficient* interactivity, scalability, and dynamic speaker selection without any "gagging" floor control. Moreover, the conference solution should mimic the face-to-face real-life conference as closely as possible and it should preferably be realisable with an existing protocol standard. In addition, the limitations that were pointed out in many existing solutions and implementations are to be addressed[2].

[2] "The scientific method, is nothing but the normal working of the human mind. That is to say, when the mind is working; that is to say further, when it is engaged in correcting its mistakes."
- Thomas Henry Huxley.

3.4 Context and Requirement Specification

Though this thesis mainly deals with voice-only conferencing, the other VoIP services should also be taken into account while designing the conference support management. VoIP services encompass many other services that drive the large-scale deployment of VoIP technology. Therefore, One-to-One call, PBX functionality, connectivity to PSTN, and Voice Mail services are also to be considered while designing the architecture for conferencing. Therefore, the conference service should co-exist with the above conventional telephony services. With this in background the following section discusses the context of the service.

3.4.1 The context of conferencing service

Primarily the conference solution is motivated by corporate environments where many branch offices of the corporate business need to collaborate with each other on a few tasks. Need for interactive conferences between these offices is felt more often than conferences with many participants spread out in Internet. Working collaboratively in this context is more useful for the distributed execution of tasks. Furthermore, traffic to and from a corporate network onto Internet can be controlled easily. Without the QoS guarantees, for high quality voice, the solution proposed in this thesis is more applicable to the corporate networks than on Internet at present. However, without loss of generality, a large number of clients that are spread across the Internet can also be considered here for conferencing (as pointed out in Chapter 1) but with some limitations and difficulty in managing such a scenario. Figure 3.1 shows an instance of three major locations that are required to collaborate with each other. Each location has an arbitrary number of participants. These locations are connected via Internet. The participants are located inside domains. These domains are logical groupings of participants and they may or may not constitute a subnet. Nonetheless, the participants in a subnet are often grouped together.

There are two parts to the VoIP conferencing setup. The front-end consists of the Client application program that runs on a participant's computer (referred as End terminal in Chapter 1). A participant uses a client program on the end terminal to avail the conference facility and VoIP services. Figure 3.1 shows many clients in those three domains. Conferencing and other services are provided through a set of servers running in the back-end. These servers use industry standard computers for supporting these facilities. They run on pre-assigned machines with known IDs, unlike clients which can run on any machine in the network. A dumb terminal, such as a hard telephone set,

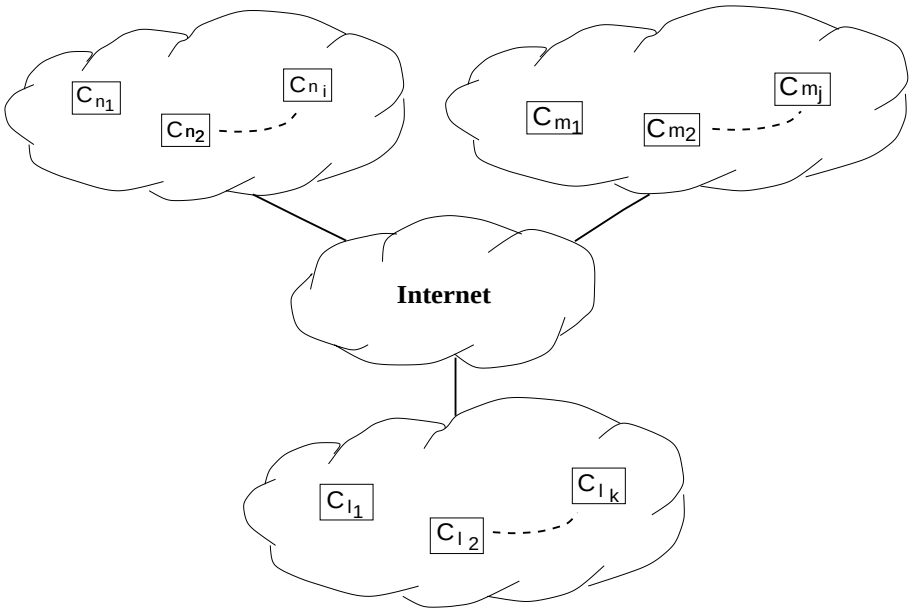


Figure 3.1: **Clients dispersed over a wide geographical area across networks**

which does not have computing power, is connected to the network through a device known as "Media Gateway". The conference support is provided using these servers and client applications, preferably using a standard protocol such as SIP.

3.4.2 Requirements

The criteria mentioned in Section 3.1 for conferencing service are the basis for the requirement specifications. Some of the requirements that are listed in this section take into account other VoIP services that are provided. Hence, some of them may not be directly connected with conferencing service. Nevertheless, an exhaustive list of requirements for this conference solution is given below.

1. Prospective clients should first "register" (usually called, *login*) with a control point to use conferencing (allowing for billing if required).

2. Participants are to be identified by their names. They must be able to "login" from anywhere. The most convenient means of addressing is the e-mail address of the participant.

3. A client should be sequestrated from managing a conference and other duties of configuring the entities in the setup.

4. Adding/deleting a client to/from an ongoing conference should be possible and simple.

5. A two party call should be peer-to-peer.

6. An on going one-to-one call should be upgradeable to a conference by adding at least a third client.

7. Many modes of building conference must be supported; Theses modes include participants joining the conferences by themselves (with password) and inviting a new user by the participant who is already in the conference.

8. The conference architecture should allow use of features such as Floor Control without affecting interactivity.

9. There must be a provision for a few participants speaking simultaneously to provide sufficient interactivity.

10. In a conference, each client should be listening to the same set of audio streams, which ensures consistency.

11. The participants should be able to have a customised mix of the streams.

12. Dynamic and seamless speaker selection should be possible without any floor control.

13. Traffic on the network due to speech and control messages should be as low as possible.

14. Scalability must ensure that a large number of clients dispersed over a wide geographical area are easily handled.

Conference architecture should consider all these requirements and provide the conferencing service as set out in the problem statement.

3.4.3 Discussion

There are two phases in providing the conferencing service; (a) conference setup and (b) maintaining and handling the audio information transfer. The former is called "signalling" in telephony. The existing telephone network uses common-channel signalling with in-band signalling, that is, the control messages and audio signals use the same logical channel. Therefore there is a multiplexing of signalling and voice signals. During the call setup phase voice will not be transmitted. For example, the DTMF code that is used for signalling, uses the same path as that of audio. The other type of signalling which is used in Internet telephony is separate channel signalling. Here the

signalling and voice are logically separated. They are sent on different logical channels. However, the logical channels may be using the same physical link.

In VoIP systems with different ports for signalling and audio, there is a complete logical isolation between signalling and audio. As signalling takes negligible bandwidth compared to the audio payload, the scalability and network traffic are mainly concerned with the audio payload.

Consequently, in this thesis, the discussions are mainly on the audio streams of conference. That is to say, sharing and transport of audio streams and selection of speakers are considered in the following discussions. Besides, the conference setup and signalling are also touched upon briefly, whenever necessary.

3.5 Architecture for Conferencing

3.5.1 Impact of requirements

Requirements 1 and 2 above lead to a control point called "Call Processor" (CP); this is like an exchange in the telephone network (and may be extended to many exchange domains and hierarchy). These requirements lead to the use of CP as an authentication point. Further, the CP, being a control point, is able to keep the mapping of the user names to the location - more specifically, the IP number - of a client.

Requirements 3 and 4 imply that calls should be set up via the CP. The CP should instruct the clients and servers whenever necessary. Multicast group address management and location of other servers are to be communicated to a client through the CP. This avoids replication of information by each client about the status of a call and other necessary information.

Requirements $5-7$ pertain to the basic VoIP call setup and conference management. The CP acts as an entity that facilitates these. The CP handles the control messages for a conference; for example, inviting a party, when a party leaves a conference, starting a conference, etc. Conferencing under central control is motivated by corporate environment and is unlike the multicast conferencing where participants easily join and leave as they wish with use of a multicast group address. For simplicity, a single CP is considered for handling control messages in the setup throughout this chapter. In fact, controlling may also be distributed (analogous to Proxies in SIP).

A conference is "built up" by adding participants successively, for example, by a moderator or a participant. Another way is to have participants contacting the controller with conference details (using, the Session Description Protocol (SDP)) and joining an ongoing conference. In this case conference details may be distributed by

the initiator/creator using means such as email, web, etc. Requirement 8 also implies the use of controllers that provide some conference floor management.

Requirements $9-12$ pertain to the quality of the conference based on interactivity (criteria $1-3$). This necessitates the selection of streams from more than one client for mixing. For this, N_{Max} clients are used. A discussion on finding the value of N_{Max} is given in Chapter 4 using *conversational analysis* of *Ethnomethodology*. A lower value of N_{Max} would facilitate the implementation of customised mixing of streams. Here in this chapter, N_{Max} is assumed to be a small number (3 or 4). In all further discussions this value is used. Requirement 12 implies the use of a new *methodology* intelligent enough to select the set of speakers to be heard by all participants in a conference.

Requirement 10 constrains this methodology to be dynamic and executed in real-time. This underlines the need for a new metric quantifying a speaker (Loudness Number of Chapter 5) that can be used to select packets from participants for mixing.

A conference is a one-to-many or many-to-many service. Hence, IP multicasting is a natural choice for its implementation. In view of possibly a large number of clients (say M) and the limited amount of mixing required, a client need not receive voice packets from all other $(M-1)$ clients. Limiting the number of streams reduces the processing burden for a client and the traffic on the network (Requirement 13). Further, in any well-behaved conference, only one speaker is active most of the time. However, there are occasions when two or more participants speak simultaneously. This makes speech less audible and requires a higher bandwidth. In order to avoid excessive traffic on the network, there is a need for filtering streams. This emphasises the need for an element in the architecture that selects an appropriate number (N_{Max}) of streams for mixing without compromising on Criterion 1. An architecture based on a new element called a "Selector" is proposed in this thesis. Selectors limit N_{Max} simultaneous speech streams using a suitable metric. This is a major step[3] towards achieving scalability by the use of *distributed* selectors (Requirement 14) keeping interactivity intact.

As shown in Figure 3.1, clients may be distributed over a wide geographical area. Thus a *distributed* mechanism is proposed for choosing the set of clients whose audio packets are to be mixed. Gordon Blair and Jean-Bernard Stefani [117] define a distributed system as follows:

> "A *distributed system* is a system designed to support the development of applications and services which can exploit a physical architecture con-

[3] "Make a clean break with your past and you will be enlightened," said the Master ... "Take the leap! You cannot cross a chasm in little jumps."

- "One Minute Wisdom" by Anthony de Mello, S J

> sisting of multiple, *autonomous* processing elements (servers) that do not share resources but cooperate by sending asynchronous messages over a communication network."

The word autonomous is emphasised here because it makes the servers work independently of other servers. It should be noted here that if there were any dependencies between servers then, tight communication between them would make the system inefficient and not scalable. Communication is only to facilitate each server getting its information from other servers.

In this architecture, each selector serves a group of clients. For example, in Figure 3.1, there could be one selector for each of the three groups. The selectors communicate among themselves to arrive at a unique set of clients whose audio streams are to be mixed. The basis for forming groups can be optimising network traffic and computation load on selectors. As each selector in a group restricts the number of packets selected for playout from that group and this remains constant, addition of a client does not cause an increase in network activity outside that group. This facilitates scalability (more discussion is in Section 3.6). The architecture comprising of these servers is described formally in the following section.

> ***Remark***: Assuming that a conference is well behaved, then there can still be packets with no speech contents transacted across Selector domains. This traffic can be avoided only if Voice Activity Detection (VAD) is incorporated. However, when each participant tries to speak to other participants in a conference (if interactivity is supported), there will be large traffic generated (overcoming VAD) that cannot be handled unless some limiting is done locally. Selectors do precisely this work with or without VAD. This architecture is driven by criteria listed in Section 3.1.

3.5.2 Architecture

There are three major software elements in the system. (i) Client or End Terminal - the front end for participants (ii) Call Processor (CP) and (iii) Selector. The latter two are servers. The proposed architecture for conferencing is shown in Figures 3.2 and 3.3. Functionally, the CP is similar to the Multipoint Controller of H.323 and somewhat equivalent to Proxy Server of SIP. Selector is somewhat similar to the Multipoint Processors of H.323. All selectors and clients are required to register with the CP or with respective CPs in their domains if many CPs are used. CP implements all the control messaging required for call set up and control. CP does not handle audio packets.

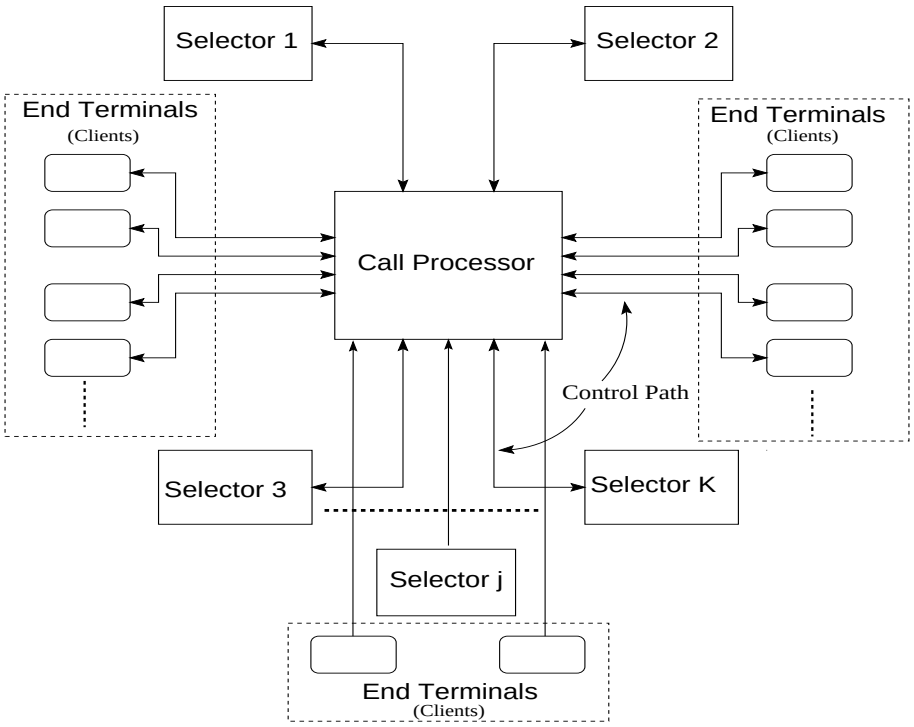


Figure 3.2: **Control path of the VoIP conference architecture.**

There are two paths of communication between the elements of the architecture. (i) Control or Signalling path (shown in Figure 3.2) and (ii) Audio/Voice path between the clients and selectors and between selectors (shown in Figure 3.3). Voice data flow is through RTP [16] which is generic and can be used at the application layer.

CP segregates clients into many groups and assigns a client to a particular selector and forms a selector domain as shown in Figure 3.3. A client is assigned to only one selector at any time. Voice path between a selector and a client is created only if the client is part of a conference. CP informs the corresponding selector whenever a client joins the conference. The selector then prepares to serve the client that has joined the conference. If the arrival of a new client brings in a new selector - in turn a new selector domain - then the CP informs all other existing selectors serving that conference. This architecture interposes only two selectors in a voice path between any two clients to reduce the total end-to-end delay while supporting interactive conversation. The functions of these elements are briefly explained in Section 3.5.3.

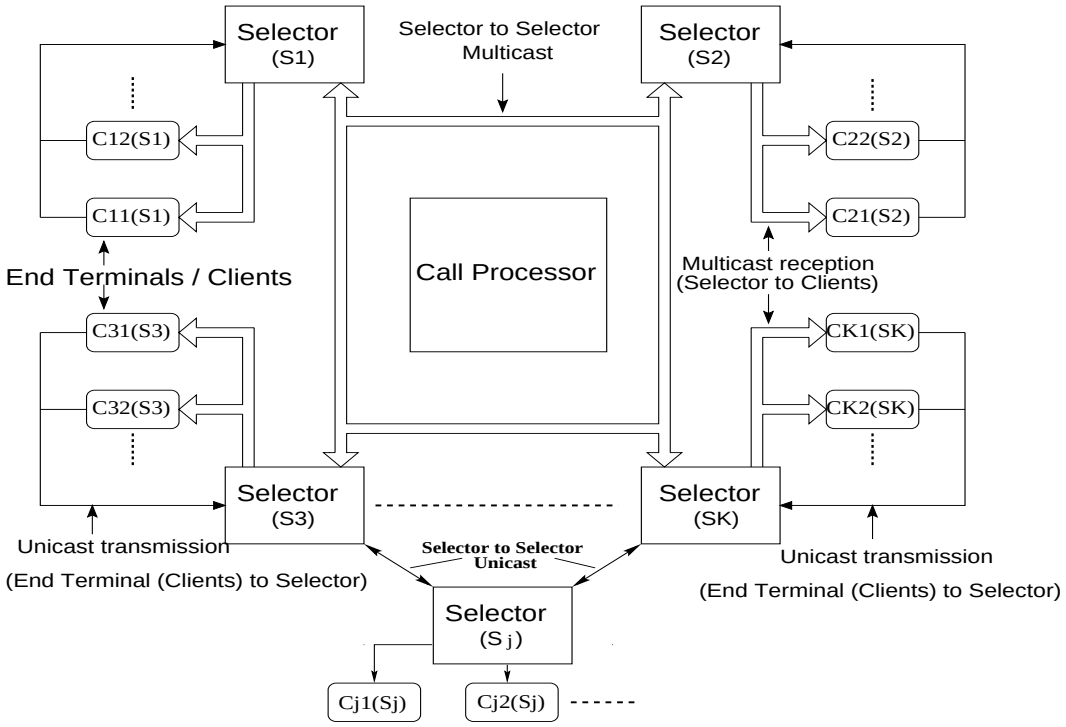


Figure 3.3: **Voice path of the VoIP conference architecture.**

3.5.3 Elements of the voice conferencing solution

3.5.3.1 Client

A Client application provides an interface for the participant to the conferencing system. It records a participant's voice, packetises it and transmits on the network. It also receives packetised audio from the network and plays it out. It may implement some proprietary techniques [81] to counter delay, jitter and packet loss problems. A user can be a part of many calls simultaneously but only one call can be live at any instant. The front end of the client software is also used for other services such as voice mail, chat, etc. A detailed explanation of the client software, which is designed as part of the prototype of this conferencing solution is given in [81].

3.5.3.2 Call Processor (CP)

Call Processor is a signalling agent. All the signalling is through the CP only. Clients have to get connected to a CP to avail VoIP services. All the selectors (and servers such as Voice Mail servers) are also connected to CP and receive instructions only from a CP. CP waits for request messages from clients on a port and acts on them as and when they are received. It provides handshake signals and connection parameters to clients

for establishing and terminating calls. It maintains a state machine representation for each call and decides a new state of the call whenever a control message related to that particular call is processed. It identifies two types of calls: a) One-to-one call and b) Conference call. Each call is represented by a Call Object which is created whenever request for a new call is accepted. Call object contains call specific information (IP addresses of the client, ports for accepting voice data, state of the call, etc.).

One-to-one call

When CP gets request for one-to-one call, it creates a call object and allocates a unique Call Identifier to that call. It also sends invitation to the destination client. If the destination client accepts the invitation, CP gives a proceed-signal to both the clients which means that the connection process is complete and a call is established. Any one of the parties can initiate call termination by sending release signal to CP. CP forwards call release message to the other client and deletes call object corresponding to that call. If any client goes on hold, the other client gets notified through the CP.

Conference call

Any one-to-one call can be converted into conference call by adding a third client to it. CP allocates a Multicast IP address and port number for the conference call. It also informs selectors involved in the conference. Any client can leave the conference. This is notified to other clients in the conference through CP. Any client in the conference can request CP to add a new client to the conference. When there are two clients in the conference, CP sends conference-terminated signal to the remaining clients and deletes the conferencing call object. However for a booked conference, the conference object exists till its scheduled time of closing or if the moderator closes it. The allocation of selectors to the clients is based on factors such as load on selectors and overall traffic saving by proper grouping. CP allots multicast addresses with the help of the proposed IETF's IGMP protocol. At present IGMP is not available in Internet.

Accounting and authentication

CP keeps logs of calls made. It also logs control messages it receives from clients and allows scheduling conferences. Authentication is done while clients login/register during the start up phase of client application.

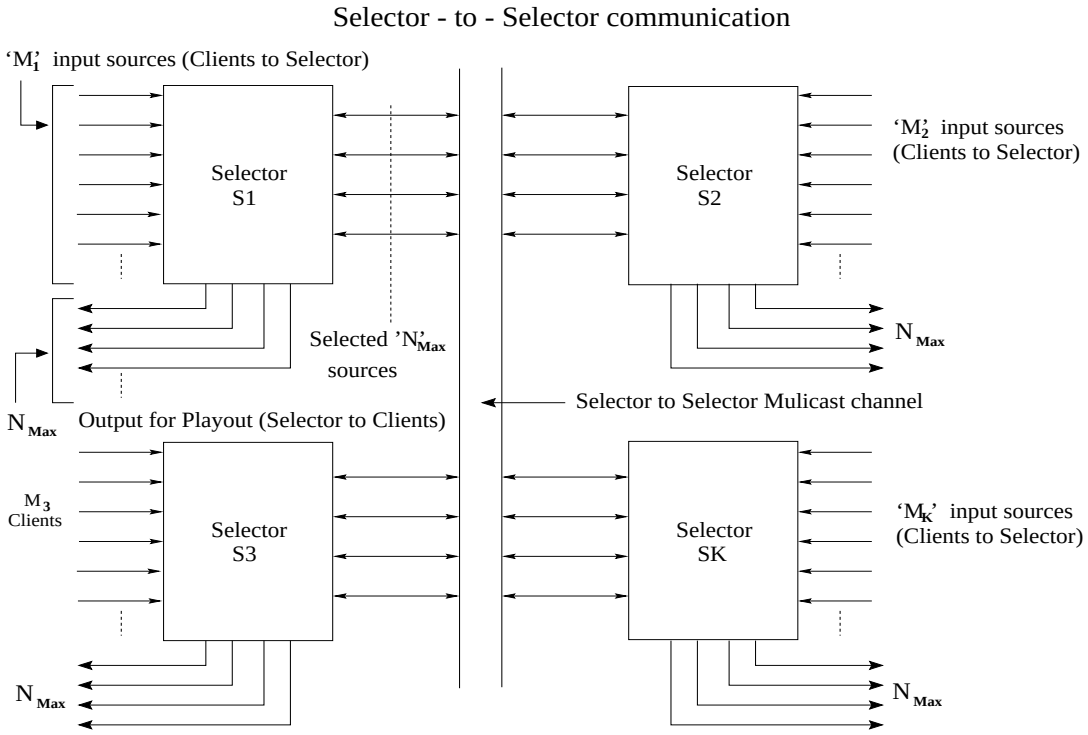


Figure 3.4: **Voice path between selectors in a typical conference.**

3.5.3.3 Selector

As proposed earlier in Section 3.5.1 a selector is used for filtering streams from each domain. It is used only for conference calls. It reduces traffic on the network by filtering out streams intelligently. Typically, one selector is used for one LAN. In order to participate in the conference clients need to be assigned one of the selectors available in the system, preferably the one which is on the same LAN. With distribution of selectors in the network, scalability can be achieved.

Clients transmit voice packets to their respective selectors using unicast transmission on a defined port of selector. Each selector is allowed to choose the "best" N_{Max} packets streams from clients in its domain. Selectors then exchange these packets using IP multicast as shown in Figure 3.4. In case a selector is behind a router, which is not multicast-enabled, then unicast is used. Since most LAN technologies support multicasting, selectors normally send selected streams to their clients on multicast as shown in Figure 3.3. However, in such cases where clients cannot receive voice packets on multicast, the selector sends unicast packets to those clients as shown in Figure 3.3. Selectors also convert audio stream coding, if necessary. After selectors exchange audio data with one another, they form a global set, say S, of clients from which the audio packets are to be mixed and played out at the clients. Finally, a selector sends N_{Max}

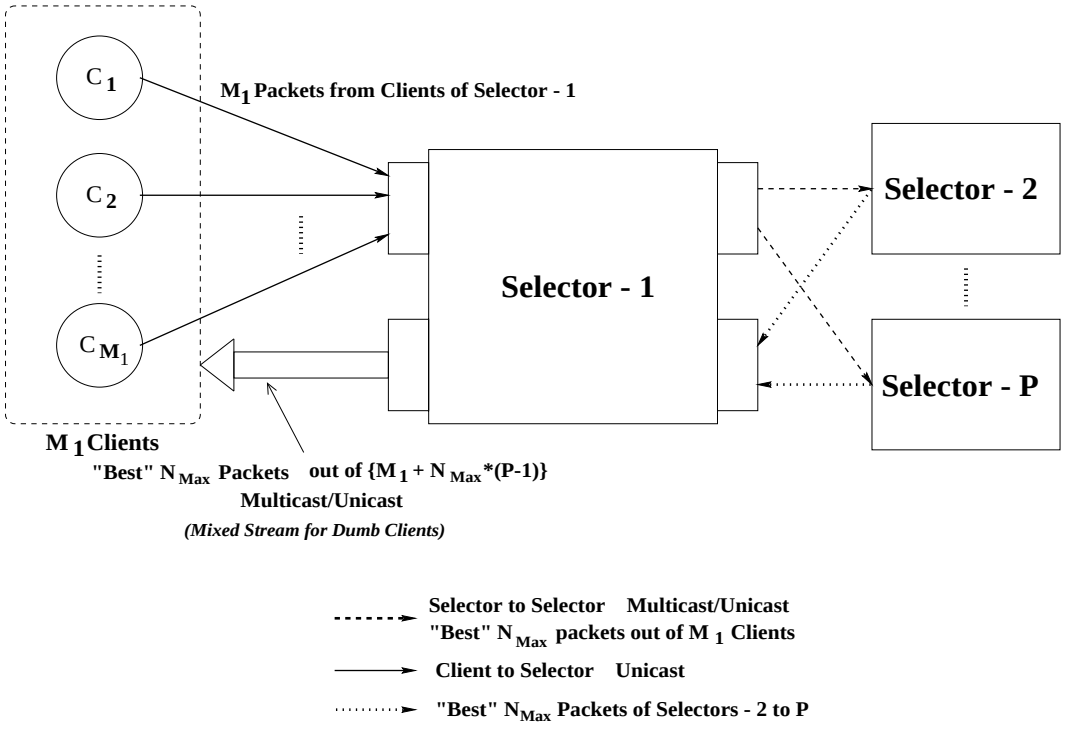


Figure 3.5: **Schematic of a selector.**

packets from the set S to the clients in its domain along with their source identification (ID). Figure 3.4 shows voice path between selectors in a typical conference.

Working of a selector

Block schematic of a selector is shown in Figure 3.5. Selector 1 serves M_1 clients, C_1 through C_{M_1}. For each mixing interval (time slot), Selector-1 chooses the "Best" N_{Max} audio packets out of the M_1 it may receive and sends these to Selectors 2 to P. The set of packets sent is labelled "*ToOtherSelectors*". In the same mixing interval[4], it also receives the best N_{Max} audio packets out of M_2 from Selector-2, and so on up to Selector-P, that is, best N_{Max} out of M_P from Selector-P. These sets of packets received from all the other Selectors, in this case Selectors 2 to P, is labelled as "*FromOtherSelectors*". The set $\{ToOtherSelectors\}$ has at most N_{Max} packets whereas $\{FromOtherSelectors\}$ has multiples of N_{Max} packets. Finally, Selector-1 selects the best N_{Max} packets from the set $\{ToOtherSelectors\} \bigcup \{FromOtherSelectors\}$ and

[4]For simplicity, propagation delay between selectors are ignored. Delay, if present, of the order of 100ms to 150ms becomes insignificant during conferencing since it is not easily perceivable. However, it can be tackled by sufficiently enqueuing the packets before mixing with the help of timestamps and NTP.

passes these packets to clients in its own domain. This procedure is carried out at all other selectors independently and N_{Max} packets are sent to clients in their domains.

The set *Best* N_{Max} *of* $(\{ToOtherSelectors\} \bigcup \{FromOtherSelectors\})$ is nothing but the set S, and it is the same at all selectors. This ensures that any client in the conference finally receives the same set of packets for mixing. Hence all clients obtain a consistent view of the conference i.e., N_{Max} out of M clients. The following simple algorithm captures operation of a selector discussed above.

Algorithm 3.1 ***Algorithm for Distributed Selection***

At Selector i, `Repeat for each time slot`

{

Step 1: *Get packets from all the clients in its domain.*

Step 2: *Find at most* N_{Max} *top streams out of* M_i *streams in its domain after arranging them in descending order based on the metric (to be discussed in Chapter 5).*

Step 3: *Store a copy of packets from those* N_{Max} *clients thus selected in a Data Base called* DB_1.

Step 4: *Send these* N_{Max} *packets to other Selectors (on Multicast or Unicast as the case may be) on the port specified by CP.*

Step 5: *Receive packets from all other Selectors and store it in Data Base called* DB_2.

Step 6: *Now compare the packets in* DB_1 *and* DB_2 *on the basis of the same metric and select top ranking* N_{Max} *amongst them (this is nothing but set S).*

Step 7: *Send these* N_{Max} *packets in set S to the* M_i *clients in its domain.*

Step 8: {`Optional`} *Mix these* N_{Max} *audio packets in set S after converting it to linear PCM and send it to dumb clients in its domain.*

}

RTP [16] is used for communication between clients and selectors. A client sends an RTP packet containing voice payload directly to its assigned selector. Selector to selectors and selector to its clients communication require multiple (i.e., N_{Max} packets) streams to be sent. RTP supports multiple packets in the same RTP payload. When a payload length becomes more than MTU of the network layers it is better to segment the large packet and send it. Timestamps and Sequence numbers are used here. Sequence number of two fragmented segments would be different but timestamps would be the same.

3.5.4 Fairness and resilience

Any distributed algorithm demands Fairness and Resilience. Definitions of Fairness and Resilience depend on the context and/or application. In the context of distributed conference application the terms are defined below [118].

> **Definition 3.1 (Fairness)** *: There is no bias in selecting a participant; as in a physical conference, a more vocal participant draws more attention.*

> **Definition 3.2 (Resilience)** *: All the clients will eventually have the same set of packets from speakers to playout after two stages of selections at selectors.*

Fairness depends on the metric that is used to select the clients. *Resilience* depends on how the selection is done. With Algorithm 3.1, assuming that there is no delay between selectors, it is easy to see that all selectors would select the same set of clients. With delay of the order of a few hundred milliseconds there would be some discrepancy in selection. Selection would eventually become same at all selectors if the metric is not varying at a rate comparable to that of delay. As both these properties are connected with the metric, discussion is deferred to Section 5.8.4 of Chapter 5.

3.6 Scalability

Distributed nature of locations of clients aid scalability in this architecture. There is saving in bandwidth because in every mixing interval, a selector sends at most N_{Max} audio packets to other selectors, irrespective of the number of clients in its own domain. The processing burden on a selector is determined by the computation necessary to identify the set $ToOtherSelectors$ in the "transmit" direction, and the set $FromOtherSelectors$ (N_{Max} packets each from other selectors) in the "receive" direction. Addition of new clients to a conference results in a slight increase in computation in the transmit direction because at a selector the best N_{Max} packets have to be chosen from a slightly bigger set. In the receive direction, computation is insensitive to the addition of new clients in any existing domain because in every mixing interval, the set S has to be formed from at most $N_{Max}d$ audio packets, where d = number of selector domains. This shows that inclusion of clients will not increase the traffic between the selectors. However, if selectors present in the system are unable to take more load a

new selector has to be brought into the system and some rearrangement of clients need to be done. This will eventually increase traffic in the system but it is inevitable.

Now the performance of Algorithm 3.1 is taken up for discussion here. Only the packets exchanged between different selector domains are considered. Selectors mostly serve the clients inside LAN. The packets that are transiting within a LAN take advantage of the higher capacity (generally coupled with multicast capabilities) and therefore are not taken into account for analysis. Moreover, WAN traffic is more important compared to LAN traffic, as it is expensive.

In a given conference let M_i be the number of clients present in domain i, where $i = 1, 2, 3, \ldots, d$. The total number of active clients in the conference is thus $M = \sum_{i=1}^{d} M_i$. As each selector from a domain will be sending only N_{Max} out of M/d packets (assuming almost equal distribution of clients) to the other selectors $M/d \gg N_{Max}$. The bandwidth used by the application over a WAN is upper-bounded by the following expression.

$$N_{Max} d(d-1) \tag{3.1}$$

Thus the total number of audio packets transiting over the WAN for each time slot is quadratic in the number of domains, i.e., bounded above by $O(d^2)$. However, it is independent of the total number of active clients. This would not have been the case had all packets been sent over the network in each time slot. For example, in the architecture proposed by Greenhalgh et al. [119], traffic grows as $O(M^2)$. The following discussion demonstrates the scalability of approach of this thesis with respect to completely distributed or full mesh case wherein no servers are used (similar to [119]) and a centralized server scheme (Section 2.3.3). Centralized server scheme is highly unlikely if the clients are distributed over a large geographical area. However, assuming that clients are very near, the centralized server scheme should aid in fixing an upper bound on the number of domains (Selector groups) beyond which the centralized scheme saves the bandwidth (other considerations for fixing the number of domains is considered in Section 3.7.1). Two cases are considered here for comparison.

3.6.1 Non-availability of multicasting between selectors

This assumption is valid in the present Internet as the network is not fully multicast enabled [61] and MBone covers only a few networks. Without any server handling the media, the number of packets in the network is of the $O(M^2)$ [119]. For centralized audio server scheme (i.e., clients contacting a server and the server picking N_{Max} packets and sending it back to all) the number of packets is $O(M)$. With the proposed approach with selectors handling clients in their respective groups, for d Selector domains, it

Table 3.1: **Order of traffic in different conference architectures**

Completely Distributed	Centralized Server	Selector Domains (d)
Unicast Communication		
$M(M-1)$	$M + N_{Max}M$	$N_{Max}d(d-1)$
Multicast Communication		
M	$M + N_{Max}$	$N_{Max}d$

goes as $O(d^2)$. Each selector is a server and in the limiting case the traffic on WAN is equivalent to the centralized server solution. Hence the traffic between selector domains can be $O(M)$. Though in the limiting case, the orders of traffic for both centralised server and selector domain is the same, the centralized server cannot scale up because of computation limitations of the server. The value of d thus calculated for this case is called as d_u.

3.6.2 Multicast enabled communication between selectors

In this case the traffic for completely distributed network is $O(M)$. For centralized server case it is $M + N_{Max}$ i.e., M for clients to server and N_{Max} in the opposite direction. With selector domains the traffic would be of $O(d)$. A simplistic view of multicasting is considered here while neglecting the topology of the network connecting the clients. Let the value of d computed for this case be d_m. Table 3.1 consolidates the traffic in the network for completely distributed clients, centralized server system and for selectors (traffic between clients and selectors is not considered here). The upper bound on number of groups, d, is calculated using this table. The first column is for fully distributed architecture with peer-to-peer communications (see Section 2.3.3) for which traffic grows at the $O(M^2)$ and is always higher than the other two. Thus distributed selector domains technique is compared only with the centralized server technique. Therefore, for unicast and multicast modes the upper bounds on the number of domains are calculated comparing only the last two columns of Table 3.1. Suppose domains become arbitrarily large, then saving would be less compared to a centralized server, and in the limiting case it can grow at the rate of M^2. Thus there is a need to find the upper bound on the number of domains in the system with which there is

a saving in bandwidth. It turns out to be $d = min\{d_u, d_m\}$. Since the calculation of number of selector domains with multicast sets too high a value ($\approx M/N_{Max}$) compared to the unicast set, to find a feasible value for the upper bound on d only the first row of Table 3.1 is taken into consideration. The inequality drawn from the table to this effect is,

$$N_{Max}d(d-1) \leq M(N_{Max}+1) \tag{3.2}$$

In the limiting case,

$$N_{Max}d^2 - N_{Max}dM(N_{Max}+1) = 0 \tag{3.3}$$

Solving for d yields

$$d = \left\lfloor \frac{N_{Max} + \sqrt{N_{Max}^2 + 4M(N_{Max}+1)N_{Max}}}{2N_{Max}} \right\rfloor \tag{3.4}$$

Firstly, for unicast network, total traffic for various values of M (with d fixed as above) is shown in 3.6(a) for $N_{Max} = 3$. Since $d \sim O(\sqrt{M})$, Figure 3.6(a) shows almost similar curves for centralized server and proposed method of selector domains. However, the values for latter is for an upper bound value of d (the limiting case, above which centralized solution is better) where the traffic in these two modes are comparable. Actually, number of domains will not be as high as given by Equation (3.4). Therefore the saving in bandwidth would be more than what is shown in the Figure 3.6(a) and the curve for Selector domains method would be below the curve for centralized server.

Secondly, assuming that the network allows multicast, Figure 3.6(b) shows a similar set of results with a significant reduction in the number of packets transported, more so for selector domains case. For distributed clients and with centralized server for audio, the number of packets in the network is $O(M)$. With the d selector groups handling the conference, the traffic on the network is $O(d)$. This shows an unambiguous reduction in the network traffic. There is a significant saving in traffic in both the cases above, i.e., with or without multicast enabled network. Figure 3.7 shows limiting values of d (Selector domains) for various values M that has traffic comparable to centralized architecture for different N_{Max}. The significance of this plot is that the difference in values of d for various N_{Max} is small. It is assumed in the above calculations that clients are distributed almost equally amongst the selectors.

It is shown here that saving is tremendous. Yet, one may contend that sending N_{Max} packets to and from all domains is a waste of resource, as most of these streams will not be selected. If just one client is active, selecting a subset of clients in that domain is unnecessary. Chapter 6 investigates alternate strategies for reducing the traffic further based on this observation.

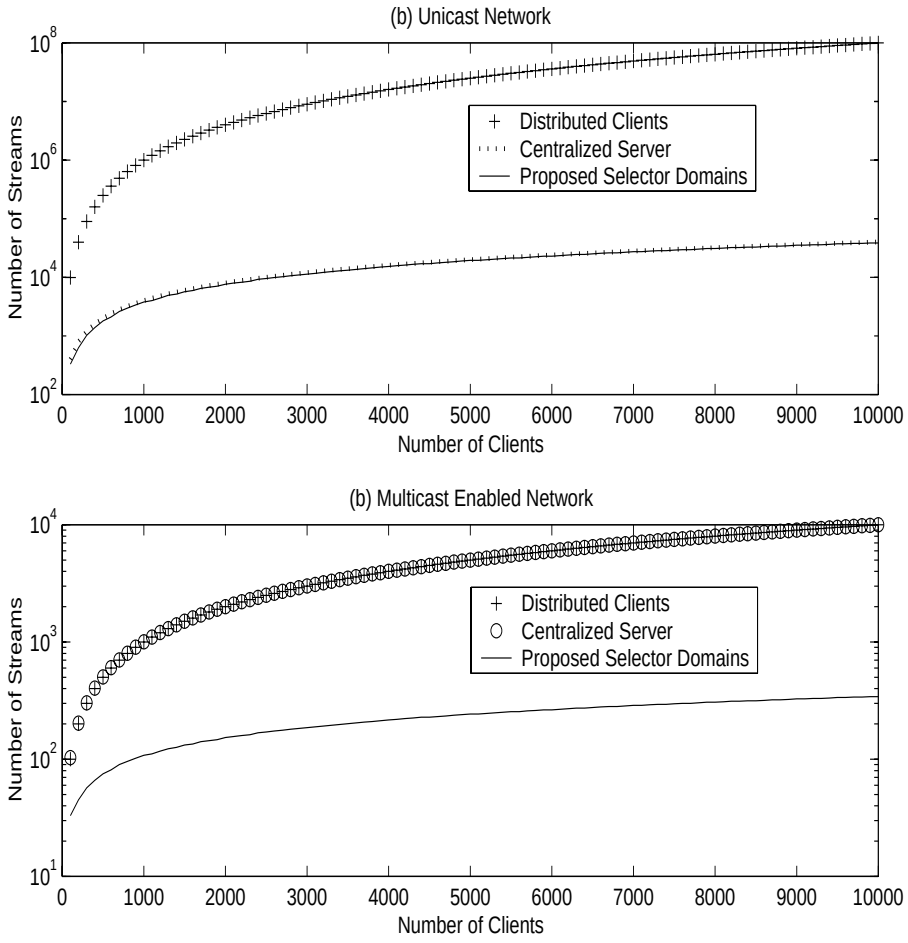


Figure 3.6: **Comparative study of traffic on the network for completely distributed, centralized server and selector domains techniques with $N_{Max} = 3$.**

3.7 Capacity of Selectors

3.7.1 Capacity of selectors versus scalability

The capacity of selectors handling streams from clients in its domain and also packets from other selectors plays an important role in designing this solution for supporting a conference. Selector-to-selector communication with N_{Max} streams exchanged needs to be taken into consideration while calculating the total number of clients that can be supported. This is because a selector has to serve its local clients as well as manage streams from other selectors.

For unicast as well as multicast each selector has to process $(d-1)$ streams from other selectors in addition to the clients in its domain. Thus having a large number of domains would increase the traffic. Depending on the number of clients expected in a

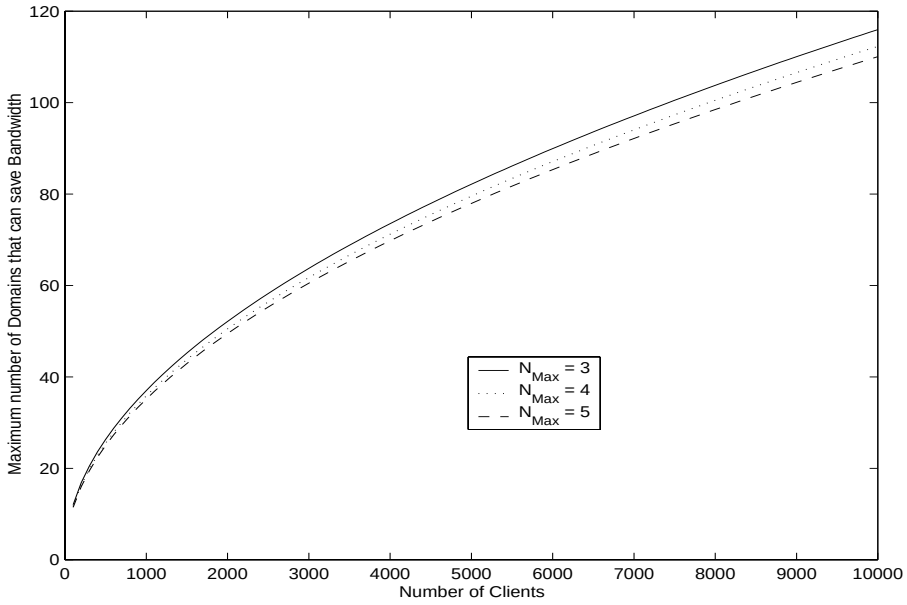


Figure 3.7: **Limiting values of d versus M for different N_{Max}.**

conference, the number of domains could be decided taking into account the capacity of selectors.

Let C_i be the maximum number of audio streams that *Selector* i can handle in period, T, which is equivalent to the time for playout of samples in that packet. This is referred to as the *capacity* of the *Selector* i. C_i is set according to the processing power of the selector. If two selectors i & j serve a conference then the maximum number of clients that can be served by both is given by,

$$M_i + M_j = C_i + C_j - 2N_{Max} \tag{3.5}$$

subject to,

$$C_i \geq M_i + N_{Max}, \text{and} \;\; C_j \geq M_j + N_{Max} \tag{3.6}$$

Similarly for three domains,

$$C_i \geq M_i + 2N_{Max}, \forall i \tag{3.7}$$

The number of domains and allocation of clients to a selector should take care of the constraint given in Equation (3.6). The optimisation problem now is to find the value of d that maximises the total number of clients served by the system. It is assumed that clients are equally distributed amongst the domains, as there may not be information to assume an apriori distribution of the clients. Moreover, it can be shown that the maximum number of clients can be served when they are distributed equally amongst

the selectors. Let C be capacity of the least powerful (weakest[5]) selector and it can be set based on a minimum system requirement a machine must meet in order to take part in a conference. This selector actually sets the number of domains as it has to handle the N_{Max} streams from each of the other $(d-1)$ selectors. Using C and N_{Max}, the number of domains d is calculated. This computed number of domains should be less than the value that was computed above considering the traffic in different types of conference implementation (centralized, etc.,). An upper bound on number of clients supported in a conference can be found given the number of active domains d and C.

For system stability - that is operation of selectors in the system without losing any packet due to overloading - any selector in the conference should be able to handle its local clients in addition to the audio packets from other domains. Then, the following inequality must hold for any Selector domain. Using generalised equation (3.7),

$$C \geq \frac{M}{d} + N_{Max}(d-1) \tag{3.8}$$

where, M/d gives the approximate distribution (assumed equal distribution for simplicity) of clients amongst the selectors.
The limiting case of Equation (3.8), considering the equality, takes the form,

$$M = (C + N_{Max})d - N_{Max}d^2 \tag{3.9}$$

To optimise M with respect to d,

$$\frac{\partial M}{\partial d} = 2N_{Max}d - (C + N_{Max}) = 0 \tag{3.10}$$

$$\text{yielding the solution, } \hat{d} = \left\lceil \frac{C + N_{Max}}{2N_{Max}} \right\rceil \tag{3.11}$$

$$\text{Then, } d = \begin{cases} \lfloor \hat{d} \rfloor & \text{if } \hat{d} \bmod 1 < 0.5 \\ \lceil \hat{d} \rceil & \text{if } \hat{d} \bmod 1 \geq 0.5 \end{cases} \tag{3.12}$$

and hence, M is obtained from Equation (3.9).

Table 3.2 gives the values of d and M computed using equations (3.9) and (3.11) for $N_{Max} = 3$ & 4. It can be seen that the values of d and M, dependent on C, are based on the least powerful selector. There is a trade-off between M and d. One can admit more domains in the conference at the expense of restricting the total number of clients M in the conference. Selectors implemented on a $Pentium\ IV\ 1.4GHz$ running Windows NT are able to handle $C = 300$ streams. Selector software requires a processing time of about 0.1ms for a packet with 40ms voice data. Therefore around 400 streams can be

[5]Strength of a chain depends on the strength of the weakest link – Anonymous

Table 3.2: **Values of d and M computed for some values of C with $N_{Max} = 3$ & 4 (d maximizing M for a given C)**

C	With $N_{Max} = 3$		With $N_{Max} = 4$	
	d	M	d	M
50	9	234	7	182
100	17	884	13	676
150	26	1950	19	1482
200	34	3434	25	2600
250	42	5332	32	4034
300	51	7650	38	5776
350	59	10384	44	7832
400	67	13534	51	10200

served on the above platform. 300 streams are safe considering OS related overheads. However, with the advent of faster computers ($\approx 3GHz$) and dedicated hardwares one can easily set C to higher values and determine d and M accordingly.

Figure 3.8(a) shows a 3D-mesh showing how the total number of clients, M, in a conference varies for selectors of different capacities and various domain sizes. The calculations above lead to maximisation of the total number of clients based on the capacity of various selectors. In Figure 3.8(b), the individual curves represent the total number of clients with respect to d and C. For a given selector capacity, there are two values of d that can support the same number of participants. The straight line in 3.8(b) demarcates the region of d (below the line) where the total WAN traffic is reduced. One can select a lower value of d on the curves (each curve is targeted for some M) below the straight line, for some capacity C to reduce total traffic on WAN. It is to be noted that the value of d in this analysis is less than the value that was upper bounded by the comparative traffic saving in selector domains technique vis-à-vis a centralized server technique in Section 3.6. The graphs presented here are for $N_{Max} = 3$.

Analysis was done with C representing the capacity of the least powerful of all selectors. Now with some of the selectors having more capacity than C more clients may be served, in turn increasing the total number of clients, M, served by the setup.

3.7.2 Conferencing with two levels of selectors

There are situations where many clients residing in one subnet overload the selector of that domain. i.e., $M_i \geq M/d$. The limit on the capacity of selector does hinder

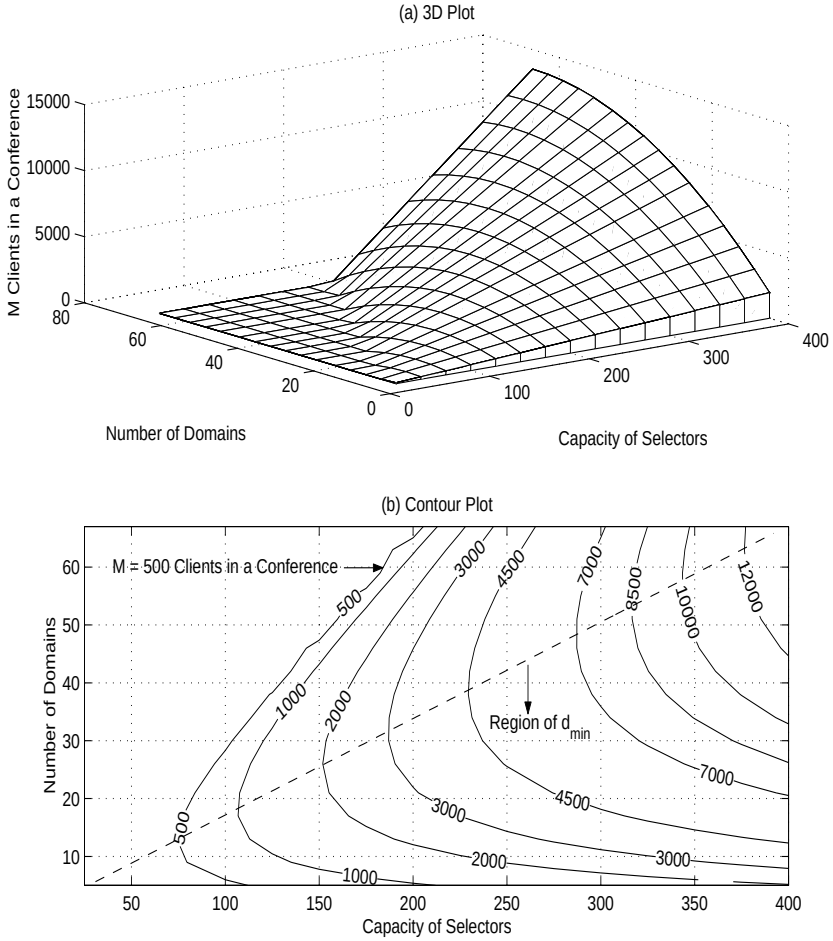


Figure 3.8: C **versus** d **for different** M **for** $N_{Max} = 3$**.**

scalability. One has to avoid the denial of service for new clients due to overloading of selectors. Further, if these extra clients are assigned to other selectors the solution is not elegant from the viewpoint of total traffic on Internet as these clients are to interact with a selector in some other network. Thus there is a need to build a slave Selector (*s-Selector*) inside the domain helping the master-Selector (*m-Selector*). Each newly created *s-Selector* must run on a separate machine. *m-Selector* can create an *s-Selector* at the cost of N_{Max} clients it is serving since, a *s-Selector* would send N_{Max} streams from its clients. With this mechanism, the *m-Selector* will be able to create utmost U *s-Selector*s given by,

$$U = \left\lfloor \frac{C - N_{Max}(d-1)}{N_{Max}} \right\rfloor \tag{3.13}$$

where C is the capacity of the *m-Selector*. One can then calculate the maximum number of clients $M_i = UC$ in each domain as well as M for the whole conference since $M = dUC$. Of course, one could further create a third level in the hierarchy, giving

the possibility of accommodating even more clients. This may be unnecessary as the number of possible clients are large enough with two levels.

3.7.3 Effect of capacity of local network

Apart from capacity of a selector, there is an important aspect that needs to be considered for finding the number of clients that can be assigned in a group. That is, bandwidth or throughput of the local network of a selector while deciding the size of Selector domains.

Let B be the bandwidth or throughput of the network/LAN connecting a selector. Let P_S is the packet size in bytes. Let each packet have T seconds of voice data. Then, size S_d of a domain can be approximated using the inequality,

$$\frac{S_d P_S}{T} \leq B \tag{3.14}$$

Thus,

$$S_d = \frac{BT}{8P_S} \tag{3.15}$$

In practice, a busy shared $10Mbps$ Ethernet network will typically supply $2-4$ Mbps (for design purposes 40% is taken) of throughput to the nodes connected to it [120, 121]. For a 40ms packet of 8bits quantisation, 8KHz sampling rate and with 40 bytes of UDP-RTP-IP header, the number of clients in a domain can be 55. This number can be enhanced with a switched network. A 10Mbps switched connection to a selector can handle up to take 138 clients. Therefore, the real bottleneck will be the link connecting selectors, as there would be streams from other selectors sending packets, which is not taken into account here. Thus, while designing the conferencing solution, the capacity (C_i) of a Selector (S_i) can be taken to be

$$C_i = \min\{C_i^c, C_i^l\} \tag{3.16}$$

where, C_i^c is the capacity of Selector S_i calculated on the basis of the hardware platform used and C_i^l is the capacity of the link connecting Selector S_i.

3.8 Features of the Architecture

3.8.1 Enhancement in quality of mixed audio

Requirements 9 to 12 (listed in Section 3.4.2) pertain to the quality of a conference. Allowing more than one (N_{Max}) speech stream concurrently amounts to permitting simultaneous speech (Requirement 9). Selectors exchange the packets and find a unique

set S across all of them allowing consistency. Transportation delay between them may introduce a small discrepancy. However this inconsistency is only for a few milliseconds which is not perceivable. As selectors send unmixed packets along with the identities (ID), it is possible to mix the packets at a client according to adjustable weights (different for each of N_{Max}) set by the user. This enhances the quality of the conference as participants can "tune" the mix to their liking (requirement 11). Use of a metric (Chapter 5) dynamically quantifying speakers would enable realtime selection of speakers without any gagging floor control. These facilities make this proposed conferencing approach more useful and imitate a face-to-face real-life conference much more closely than existing solutions.

3.8.2 Setting up a multicast tree

If multicast is enabled in Internet, then selectors can exchange packets by multicasting. Selectors may act as proxies for clients in their domain, therefore multicast tree can be built only with selectors as nodes. Hence the architecture circumvents setting up or changing multicast tree structure whenever clients join or leave the conference. Selectors can also work as an "allowed" node behind the Firewalls. Furthermore, Rosenberg and Schulzrinne comment in [61] that MBone or multicasting at present is nowhere in sight for the use by Internet community. Thus it would be appropriate to minimise the number of Selector domains and allowing unicast between them; this would still result in reduced traffic. The same architecture works when multicast is available partially. This can be achieved by allow multicast memberships to those selectors which are on MBone (or multicast enabled network) utilising multicast capability. They can send copy of packets on unicast to those which are not on multicast, thus the benefit of multicast may be reaped as much as possible.

3.8.3 Scalability

The advantage of distributed architecture is resource sharing, availability of resource, extensibility, and performance due to parallelism. It is clear from the discussions above that the architecture with selectors leads to a significant reduction in processing requirements at clients, as well as communication costs on the WANs. The architecture is distributed and audio packets have to go through at most two selectors, hence delay is minimised compared to mixing in each nodes of tree architectures [40, 41, 42, 27, 39].

Selectors may always use multicast inside the domain, as almost all present day networks allow multicasting inside subnets, thus reducing traffic further. When multicast

is not supported between domains, there can be application level multicasting similar to what has been discussed previously. This calls for some information regarding the topology of the network.

3.8.4 A discussion on features

Some extra processing is however required at each client or at selectors in case of dumb clients to compute the metric to harness the advantage of this architecture. This metric is a necessity to decide which of the clients are to be selected for mixing and playing out. Thus the computation complexity of the metric calculation should be of linear order or sub linear. Saving in bandwidth (traffic) as well as preclusion of gagging conference controller is at the cost of an extra computation of the metric to be explained in Chapter 5. However, computation of this metric is assumed to be simple.

Solutions proposed in this thesis are at the application layer (loosely coupled with the lower layers such as UDP/IP), features such as saving bandwidth, improvement in the quality of the conference, etc., are achieved, to an extent, independently of the lower layers. This architecture can further improve quality of the conference service if and when QoS guarantee is available. The technique of placing more intelligence or functionality at the application layer is called as Application Layer Framing (ALF) [122]. ALF assumes that the best way to meet diverse requirements is to leave as much functionality and flexibility as possible to the application.

There are situations where a client is alone in its domain without a nearby selector to serve. In this situation the client has to be assigned a to a selector of another domain. To save bandwidth this client may receive a mixed stream rather than individual streams. Further, when all the selectors are working at their maximum capacities, a new client added to a conference brings in a new selector. This results in increased traffic, which is inevitable.

3.8.5 A remark on echo

During a conference the voice stream of a speaker comes back to the client. It is played out by the participant's terminal resulting in an echo that at times is annoying. It is due to selectors selecting the N_{Max} speakers and sending it back to all clients. Note that this echo is different from that of network or hardware induced or acoustic echo. H.323 recommendation leaves this problem to be solved by application developers.

In this setup, since the streams are not mixed at the selectors, a client's own packet can be identified and removed before mixing. Nonetheless, it is recommended that a

client's own stream should be mixed but with less weight compared to other selected participants' speech. This gives a "sidetone" to the speaker and confirming that all others are hearing him/her. This facility is enabled using the "Self" bar shown in Figure A.6 in Appendix A. In case of dumb clients, selectors have to mix the packets and send it back. In this case selectors have to form $(N_{Max} + 1)$ different streams. While sending mixed packets to a client, that client's speech is given very low weight such as 5-10%. This avoids annoying echo yet useful in providing "sidetone".

3.9 Limitations

This chapter deals with a distributed architecture for providing VoIP conferencing on Internet. The assumptions that were made here convey a simplistic view of the network and distribution of the clients in the network. The analyses are flaccid in dealing with the compact requirements that are necessary to be directly used. The study does not make use of what all a distributed architecture can cater to. The assumptions of allowing only N_{Max} simultaneous speakers and the need for a metric that works in realtime to assist in selecting speakers to be heard are not consolidated. Without suitably fixing N_{Max} and an appropriate metric this architecture would be of little value. The architecture and its analysis are subjected to these assumptions.

3.10 Conclusions

The proposed architecture satisfies all the criteria listed in Section 3.1. Moreover, the requirements listed in 3.4.2 are all addressed while designing the architecture. This architecture does not have any floor control that are felt by many [27, 29] as gagging for the participants.

The architecture allows a novel way of implementing a 'controller less but dynamically controlled' conference offering interactivity and dynamic speaker selection. However, it has the innate *risk*[6] of working with two assumptions pointed out above while meeting the requirement specifications. They are, (a) the value of N_{Max} is considered to be as low as 3 or 4, (b) there is a metric which can facilitate smooth conduct of a conference while providing sufficient interactions for the participants. The issue of arriving at a value of N_{Max} is resolved using *Ethnomethodology* in Chapter 4. Finding a metric, with the desired attributes, is addressed in Chapter 5.

[6]It is no accident that the word "risk" in the original Arabic meant 'earning one's daily bread'
- Peter F Drucker, quoted in Jack Beatty's 'The world According to Drucker'

In the problem definition section, the stress was also on architecture complying with any of the existing standards. CP and selectors can be implemented with IETF's SIP [98]. Proxy Servers of SIP along with many other servers may carry out the responsibilities of a CP. Conference Servers defined on the lines of User Agents (UA) may execute the necessary functions of selectors. This chapter covered only the vistas for a useful architecture. An in depth study on these aspects are deferred to the chapter on implementation (Section 8.5) where they are more appropriate in the light of discussions on implementations.

Chapter 4

On Fixing the Number of Floors

4.1 Introduction

In the previous chapter the need for reducing the number of simultaneous speakers was seen from the point of view of reducing the network traffic and communication cost. In addition, it was argued in Chapter 2 that there is a need for simultaneous speakers to provide a feel of a face-to-face real-life conference. It was not stipulated as to how many simultaneous speakers are to be allowed; N_{Max} was not fixed there. It was taken to be a small number such as (3 or 4) for discussion. In this chapter an argument is evolved and a qualitative study involving listeners is presented to specify N_{Max}. In the qualitative study, opinion of listeners is sought. They were asked to identify the original speech whose transcript is known and is mixed with many other participants' speech. An upper limit on the number of simultaneous speakers in a conference is aimed at using the contributions of social scientists Sacks, Schegloff and Gefferson [123] dating back to 1974. As there cannot be too many participants speaking (addressing all participants) simultaneously, finding an upper bound on the number of simultaneous speakers, N_{Max}, is an important step.

As network connectivity and bandwidth improves, exploring ways to support the collaborative interaction of distributed participants has become rigorously pursued topic of research and discussion. Increasingly powerful systems for desktop conferencing, group authoring, and distributed design promise to fundamentally change the way participants in modern society interact with each other, both casually and in formal business contexts.

As a matter of fact, the pace of technical advancements in products has raced far ahead of identification and development of measures and techniques for characterising and evaluating the novel communication environments. Many projects and designs mainly focus on the bandwidth, frame rates, client-server designs, tight control, etc.,

and invariably assume more-the-merrier attitude. It is assumed that higher bandwidth, more facilities, and more complex control mechanisms lead to better quality of interaction. These are some, but not the only, means for improving interaction. This approach ignores the functional utility of the environment that is used for collaboration [124, 24]. Eckehard Doerry [124, 24] criticises this approach by saying, "It is keeping *form* before *function*". This thesis takes an approach that considers both aspects, *technical* and *functional*. In addition, it tries to use both of them together, so that there can be a different approach towards solving problems, sometimes resulting in the path of least effort. Thus it is appropriate to investigate the utility of Conversational Analysis (Section 4.5) that is a part of the functional aspect of the audio conferencing service.

Conversational analysis is taken up for two reasons: initially for fixing the number of floors and later for aiding in framing a simple floor control. It is assumed that the underlying network provides sufficient bandwidth for the application. The concern in this chapter of the thesis, consequently, is not as specific to *utilising* the available bandwidth as it is to *designing* the application over the network that efficiently uses available bandwidth. Conversational analysis has been used effectively by Dingley [125], also in ascertaining whether commercially available synchronous messengers can support real world ad hoc interactions. Before examining the conversational analysis, the characteristics of a voice conference are examined in the context of Computer Supported Cooperative Work[1] with other media types.

4.2 Computer Supported Cooperative Work

In today's shrinking world, collaborative work is important in business, research and many maintenance activities. The main facilitators for this are computers, computer networks and Internet. Such collaborative work is termed Computer Supported Cooperative Work (CSCW). Several advantages of CSCW with audio conferencing have been detailed earlier in Section 1.4 and they have motivated this work. Packet networks such as Internet have their intrinsic advantages and limitations in the context of transporting different types of traffic such as data, audio and video. The requirement is to build a CSCW application that supports audio conference that mimics acceptably closely a face-to-face real-life conference. In audio conferencing over Internet, it is desirable that audio from only selected speakers be broadcast/multicast to all participants.

A *conference* is defined as a virtual entity that facilitates realtime interactive data/voice/video transmission from "active" participants to all participants. A confer-

[1] This was informally referred to as *Collaborative Work* in Chapter 2.

ence may be classified as, inter alia:

1. One-To-All (OTA), for instance, like in a radio broadcast wherein the person who speaks is pre-specified.

2. Many-To-All (MTA), where possibly more than one non-specific speaker is permitted at a time to address all participants.

3. Some-To-All (STA), wherein a select few participants are permitted to speak; the rest are listener-only parties. In STA the set of speakers who can speak is fixed where as in MTA any one in the conference may speak to others.

4. All-To-All (ATA), where the maximum number of concurrent speakers may be equal to the number of participants, *a la* concert on the network.

Audio conference studied here falls into the MTA category. Different types of media used in conferences are considered below for comparison. The main resources in any collaborative work being textual or audio-visual communication, information exchange using these media for CSCW are briefly discussed.

4.2.1 Textual communication

A conference may involve only textual mode of communication such as chat and email list. Text includes ASCII files, spreadsheets, WWW pages and coded files. Text can be stored and shared using editors, chat programs. It can be easily shared with others in a CSCW. Time[2] and Space[3] constraints are not significant as storage space, display area and high speed communication requirements are not the bottlenecks in the Internet for information exchange in textual form. Also, there is no realtime requirement for text based information exchange, except for a non-critical upper bound for delay in chat applications like IRC and ICQ. It can accommodate a reasonable number of participants without serious impediments. Many participants can be typing in and there is no need to control who should write next as it can accommodate all participants. Users have identified some problems associated with text based conferencing, a detailed account of which is given in [126]. An example of turn-taking analysis of the text based conference and, in general, Computer Mediated Communication (CMC) is dealt with in [127]. A

[2]Time in the sense of realtime constraints or interactiveness as in the case of audio chat.

[3]Space in the sense of the area that a medium occupies in terms of visual area of a monitor or the storage capacity of the computers.

recent study and proposal for improvements in text based conferencing that takes into account some of the turn-negotiating, social presence, and activity facets of a face-to-face conference is found in [128]. Threaded chat application for turn completions is reported in [129].

4.2.2 Audio-Visual communication

The resources under this category are, speech, sound, images and graphics. Image and graphics may be still or moving. There is a differentiation between realtime and non-realtime applications with respect to these media. Non-realtime applications may use these resources as files and in turn may be classified under "Text" for all practical purposes. Therefore only realtime Audio-Visual media for the CSCW are considered here. In fact a realtime video most often would have audio also. Supporting a face-to-face real-life like audio-visual conference on network where tens of participants are present is a difficult task.

In an interactive CSCW resources are to be shared in realtime. This requirement is one of the main bottlenecks in designing such applications, assuming that the network is capable of sufficiently providing the required bandwidth. Limitation of time and space is the main hurdle. To bring out this hindrance clearly, an example of a real-life face-to-face conference where tens of participants may be present is taken up. In such a conference when one person is speaking to the participants there can be interruptions and cross talks. All can hear these voice signals. Physical gestures can control the participants automatically. When the same service (Audio-Visual conference) is implemented on the Internet space limitations come to fore, as pictures from all the participants' terminals cannot be accommodated on a computer screen. Yet there can be a limited number of smaller windows or thumbnails on the screen showing the participants. Multiplexing pictures may solve this issue to some extent. Here multiplexing means many participants using the different areas of space (screen). In addition, screen can be shared at different times, i.e., multiplexing with respect to time.

However, for realtime audio, if more than one participant is speaking at the same time, then their streams have to be *mixed* before playing out audio at the participants' terminals. During multiplexing, media/resources except audio occupy different slots in time. In case of audio, multiplexing cannot solve this problem, as the resource is *time* itself. The capability to undergo mixing and yet be very useful (mixing is required for reasons that follow in later sections) is a fundamental characteristic of audio. Therefore, there is a requirement for some form of authentication, called "floors" which is formally defined later in this chapter, and it is a common entity for any CSCW. Two examples

of existing conference solutions are given below.

Solution A: Interactive Remote Instruction (IRI)

An example of Interactive Remote Instruction (IRI) developed by Old Dominion University is reported in [47, 130]. In this project two way audio and one video stream is used. Floors (tokens to speak) are allocated on the basis of who reserves it first. Thus there will be some request for transmitting audio and transmission can take place only if the floor is allotted. The time for holding the floor is dependent on the number of clients waiting for their turns [47]. Distributed Mutual Exclusion algorithms developed for the IRI have also been reported in [47].

Solution B: MASSIVE

MASSIVE [131] is an audio-visual and text based conferencing system developed by Chris Greenhalgh and Steve Benford of the University of Nottingham. It aims to support multiple participants, and heterogeneity in participants' media. They proposed *Spatial Mediation*, which means that loudness and eye contacts should enable the Balance of Power (i.e., who can speak). The design was aimed to be able to work on Wide Area Network and to be scalable as well.

Scalability, is based on the concept of *aura*[4] as originally used in the DIVE system [103] using multicast. Each object in a virtual world has an aura for each medium – visual, audio, text, etc. – in which it can interact. This aura defines the volume of space within which interaction is possible: interaction between two objects becomes possible only when their auras collide or overlap. Actually, the visual scenes of real world are mapped and a user moves in that synthetic space. Exactly what happens when auras overlap depends on the details of the system architecture and will essentially involve establishing a network connection between objects or an exchange of addresses or references. Turn-taking mechanism observed in MASSIVE is dealt in detail in [133, 134]. They also propose user embodiment with representative body images for better turn-taking for participants.

There is no attempt to find the number of floors required in both the above implementations. MASSIVE implicitly assumes that any number of participants colliding in an aura would be able to interact. IRI assumes some control. Also, many other applications such as DIVE, assume some controller for facilitating different participants to speak mutually exclusively in time [47, 48].

[4] *Aura*: A field of influence surrounding a person/object [132].

Therefore it appears that, the issue of fixing the number of floors is simple and may be addressed flexibly as required by the application. The next section deals with the motivation (apart from the need dictated by the architecture discussed in Chapter 3) for addressing this issue. It is envisaged that the solution to this issue may be adopted into the standards.

4.3 Motivation for Fixing N_{Max}

The ITU-T standard H.323 [91] mentions selection of N out of M participants in a conference but it does not set the value for N. The onus is on the application developers. In SIP there is no defined standard architecture and control except an Internet draft for conference control [135] for tight conferences. However, SIP provides some flexibility to the application developers in implementation of a conference service. Both these standards do not specify the number of floors or even a bound for them.

There are many detailed studies [45, 46, 87, 86, 48] on floor control for a CSCW, but surprisingly, there has been no attempt to specify N. This may be due to the implicit assumption that only one person speaks at a time. As discussed earlier (Chapter 2), it becomes gagging for the participants when they have no spontaneity. Consequently, conferencing solution would be blamed for lack of quality however fair or smart the controller may be[5]. Thus it is desirable to stipulate an upper bound on N (i.e., N_{Max} of Chapter 3) for a voice-only conference. Fixing N_{Max} will in fact assist future application designs to tread a well-defined path unlike H.323, which leaves fixing N to the future revisions of the standard. More significantly, fixing N would in fact help in defining conference architectures.

4.4 Problem Formulation for Finding the Number of Floors

The definition of *Floor* of a conference is given below.

> **Definition 4.1** *Floor is a virtual platform (as in any shared system), which a participant must necessarily have access, though temporarily, to obtain permission to transmit voice.*

[5]Gilbs's Law of Unreliability:
At the source of every error which is blamed on the computer you will find at least two human errors, including the error of blaming it on the computer.

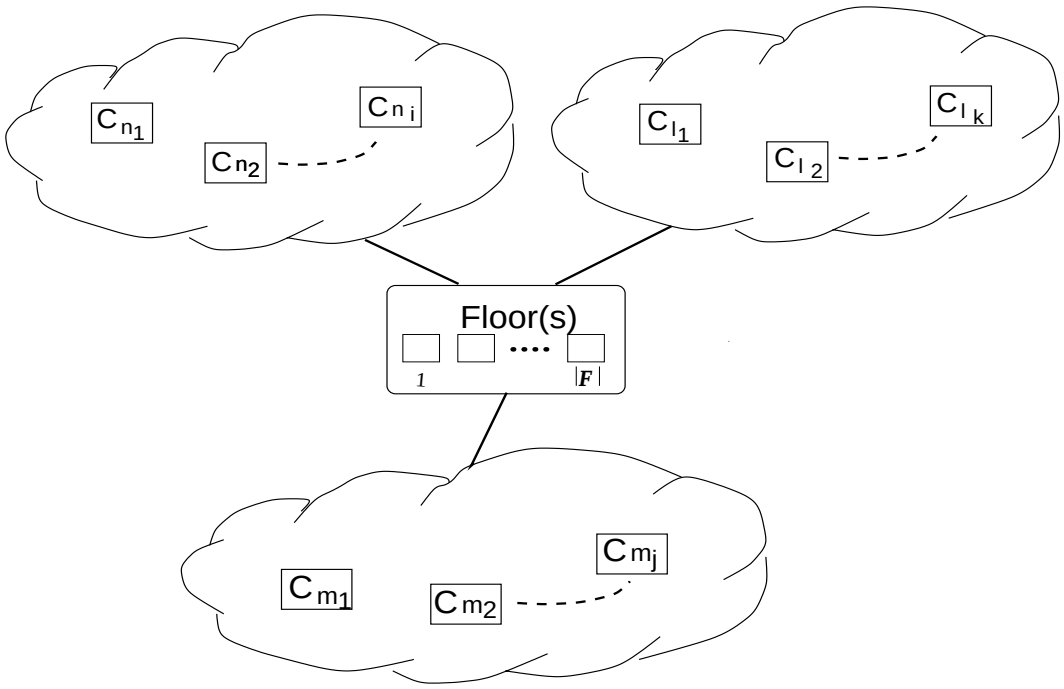


Figure 4.1: **Participants sharing multiple floors**

A floor can also be thought of as a privilege in the domain of audio part of a CSCW to mean "the right to speak". A floor F is a tuple

$$F = \{F_{id}, R, U, T_s, T_e, A\} \tag{4.1}$$

where F_{id} is a unique floor identifier within the shared workspace for a resource R, assigned to a user U starting at time T_s, ending at time T_e and with a list of floor attributes A [44]. For a conference there can be more than one floor, $|\mathcal{F}|$, where the set of floors $\mathcal{F}$ is shared by all participants (Figure 4.1).

Let Ω be the set of all participants. Then $M = |\Omega|$ is the number of participants. $S \subseteq \Omega$ is the set of participants (Set S was introduced in Section 3.5.3.3, used hereafter with equivalence to floors i.e., $|\mathcal{F}| = |\mathcal{S}|$) permitted through a token to access the floor. Other participants do not hear audio from a participant without a token. With small M, $S = \Omega$, or $|S| = M$, may be feasible when the service is ported on packet networks for a CSCW. That is, every participant has a token. However, as M becomes large, typically tens/hundreds, it is not pragmatic – in fact, not even necessary and desirable among well-behaved participants – to have $|S| = M$. Hence the number of tokens, $N = |S| < M$. The "number of tokens" is also referred to as the "number of floors". Having a token is equivalent to having a floor.

There are two situations which merit concern in allotting floors. On one hand, if S is static, which is the case when the participants in S are time invariant, the

implications are that not every participant has a floor and floors are not transferable among participants. On the other hand, if S is dynamic, which is the case when the participants in S are time variant, the implication is that not every participant has a floor; the floors are transferable. An interactive conference falls into the latter case where S is time variant. This calls for some form of floor allocation/control. Here again, there are two issues to be considered.

A. Specifying N, the number of floors. This chapter addresses the issue of fixing an upper bound on N, that was referred to as N_{Max} in Chapter 3 and N_{Max} is used further in place of N.

B. Managing these N_{Max} floors to ensure "fair" floor sharing when there is a conflict. Chapter 5 addresses this issue of selecting speakers by quantifying their activity in an audio-only conference.

Before addressing the problem of fixing floors, there is a need to understand the underlying concerns that dictate the acceptability of a solution. For this reason, conversational analysis is presented in the next section sufficiently comprehensively. Solutions are taken up in Section 4.6.

4.5 Conversational Analysis

Conversational analysis is an analytical method developed [123] within the discipline of ethnomethodology[6] as an evaluative tool for CSCW characterising the communicative efficacy of computer-mediated communication environments. It is based on epistemological[7] foundation known as situated action, which stresses the knowledgeability of actors and how they use common-sense practices and procedures to produce, analyse and make sense of one another's actions and their local or situated circumstances [124]. In practice, conversational analysis documents ways in which participants maintain moment-by-moment coherence of conversation in response to the contingencies of local context. Traditionally, conversational analysis has been used extensively as a documentary tool, providing empirical evidence of how participants collaboratively regulate their interaction. In the context of VoIP conference in this chapter, ideas from

[6] *Ethnomethodology*: The branch of sociology that deals with the codes and conventions that underlie everyday social interactions and activities.

[7] *Epistemology*: The branch of philosophy that studies the nature of knowledge, its presuppositions and foundations, and its extent and validity.

conversational analysis are used for fixing the maximum number of floors. Conversational analysis is used here not just as a documentary tool but as the basis for the arguments that ensue to authenticate the results.

4.5.1 Turn-taking

Sacks, et al. [123], proposed for the first time a detailed analysis of conversations. Their paper is taken as the basis for some of the ideas developed in this thesis. They claim that their analysis of conversations has even cross-cultural validity. A useful aspect of any conversation is *turn-taking*. Turn-taking is used for ordering in games, customer service, etc. It is also used in *Speech Exchange Systems* such as interviews, meetings and debates. It provides an opportunity for a participant to capture the attention of others in a conversation. Thus it is a prominent type of social organisation [123] and interaction. The implicit meaning and importance of turn-taking can be qualified by quoting Sacks, et al. [123].

> "For a socially organised activity the presence of 'turns' suggests an economy, with turns for something being valued and with means for allocating them, which affect their relative distribution, as in economies".

This meaning of turn-taking is consistent with the idea of floors as a resource, which is shared and limited, expressed earlier in this chapter. In this sense, following *facts*[8] for any conversation are noted [123].

1. Speaker-change recurs or, at least, occurs.
2. Overwhelmingly, one party talks at a time.
3. Occurrences of more than one speaker at a time are common, but brief.
4. *Transitions*[9] from one turn to a next with no gap or no overlap are common. Together with slight gap or overlap makes up majority of transitions (see Figure 4.2 which is similar to the one in [45]).
5. Turn order is not fixed, but varies.
6. Turn size is not fixed, but varies.
7. Length of a conversation is not specified in advance.

[8] As discussed in [123].

[9] When there is a change in speaker.

8. What the participants say is not specified in advance.

9. Relative distribution of turns is not specified in advance.

10. Number of participants vary.

11. Talk can be continuous or discontinuous.

12. Turn allocation techniques such as a current speaker selecting the next speaker or self-selection of speaker might occur.

13. Various 'turn-constructional units' are employed; and they may be a word or a sentence in length.

14. Repair mechanisms exist for dealing with turn-taking errors and violations; e.g., if two participants find themselves talking at the same time, one of them will stop, thus repairing the error.

The detailed discussion of these are also provided in [123].

4.5.2 A model for turn-taking

A model that is compatible with the above set of observations of a conversation was proposed in [123]. The model aims to capture the conversation on the basis of turn-taking mechanisms. The model has two components and a set of rules. They are paraphrased from the seminal paper of Sacks, Schegloff and Jefferson in 1974 [123].

A. Components:

A1. Turn-Constructional Component:

This component helps in starting a transition and hence a turn. It is mainly the starting of an interruption while the present speaker is speaking or during a pause or silence. There are many unit types – sentential, clausal, phrasal, etc. As for the unit types, which a speaker employs in starting the construction of a turn's talk, the speaker is initially entitled in having a turn to one such unit. The first possible completion of a first unit constitutes an initial transition-relevance point. Transfer of speakership is co-ordinated by reference to such transitions.

A2. Turn-Allocation Component:

Turn-Allocation techniques are distributed into two groups:

1. *Next turn is allocated by the current speaker*: Here the moderator or the one who is speaking would (s)elect the next speaker. It is observed in [133] that these transitions are smooth. But in a conference when voice is the only channel used, this can not be ensured always. Thus the next type of turn-taking also ensues.
2. *Next turn is allocated by self-selection*: Here the next person who speaks is selected by oneself. The transitions here are marked by the prevalence of long silences compared to the previous method. Also, sometimes there are breakdowns in the conversation due to collisions. Bowers [133] calls this a disfluence in turn-taking. This requires some corrective measures called repair.

B. Rules

The following are the rules that govern turn construction. They are mentioned here for completeness as these simple rules apply to any conversation.

1. At a transition, when the current speaker reaches a point at which a turn can be taken, one of the following three events happen:
   a) If turn-so-far is constructed as to involve the current speakers selecting the next then the selected party has the right to take the next turn.
   b) If current speaker does not select next speaker then self-selection may be instituted or the first starter acquires the right to turn.
   c) If current speaker does not select next speaker then current speaker may, but need not, continue, unless another self-selects.
2. The above rule applies at the next transition relevance point if there is no transition either by next selection by the present speaker or self-select and, if the present party continued.

C. Consequences

The consequences of this model are:

1. *There is an intrinsic motivation for listening.* This consequence is reasonable in itself as when persons speak there is an intrinsic expectation (by themselves) that they want to exchange their ideas or thoughts. Also, a

turn obliges any willing or potentially intending speaker to listen to and analyse each utterance across its delivery. Thus a participant willing to speak next or selected to speak next needs to listen and understand. In fact, without this a serious or an intelligent conversation would not exist in the first place!

2. *Turn-taking organisation at least partially controls the understanding of the utterances.* Turn-taking happens in a conversation so as to enable the other participants to get a chance to speak, without which there would be no meaningful conversation. A conversation without turn-taking would be a broadcast (like in radio) and conversation would not exist. Since a major class of turn-taking is 'current selects next' which constitutes 'first pair-parts'– i.e., type characterised utterances such as greeting, question and complaint – the next speaker has to 'answer' these intentions by speech. Thus turn-taking is part of any conversation which may not control the understanding completely but does so at least partially.

Furthermore, turn beginning is an important initial place, and an important initial resource for the projection of turn-shape or turn-type that overlaps the previous turn which is recycled [136]. The next part of conversational analysis is the mechanism of repairs that ensue during a conflict between speakers in a conversation.

4.5.3 Repairs in a conversation

In a conversation 'Organisation of Repair' addresses the recurrent problems in speaking/hearing/understanding. Also, when more than one participant speak simultaneously, an overlap occurs. 'Interruption' is the start of the second speaker while the first speaker is speaking. Overlapping talk is a recognizable event which leads to repairs in a conversation.

Some properties of overlap are noted by Schegloff [137]. The important ones are: (a) most overlaps are over very quickly; (b) some overlaps persist to considerable length; and (c) many overlaps are the site of hitches and perturbations in the production of the talk. Overlaps are resolved (i) after one beat – a small unit of time equivalent to an accent time, or (ii) after competition if they are longer. Overlapping leads to repair of the defunct conversation and a repair in turn sets up turn-taking.

Repairs are some practices to deal with the understanding of the conversation. They are composed of two parts – repair *initiation* and repair *outcome*. The distinction in initiation is made by *who* initiates the repair and *where* it is initiated. The outcome is

the corrected[10] conversation or even abandonment (failure). There can be self-initiated repair or others-initiated repair.

Self-initiated Repair :

This normally involves the speaker of the trouble-source initiating repair and executing it to conclusion in the same turn [139].

Others-initiated Repair :

This generally involves a recipient of the problematic talk initiating the repair, but leaving it for the speakers of the trouble-source to deal with the trouble themselves in the ensuing turn [139].

Self-repair and other-repair are significantly studied in the literature [138]. Some of the properties of repairs are [138]: (a) Self-repair can issue from self-initiation or other-initiation; (b) Other-repair can issue from self-initiation or other-initiation; (c) Failures can issue from self-initiation or other-initiation; (d) Self- and other-initiated repairs have regular, and clearly different placements relative to the troubled source and they are done with clearly different initiator techniques; and (e) Self- and other-initiated repairs deal with same trouble types and they are organised by reference to each other.

Opportunities for self-initiation repair come before opportunities for other-initiation. Other-initiated repair also yields self-correction. Self- and other-repairs are not alternatives but organisation of repair in conversation provides centrally for self-correction which can be arrived at by alternative routes of self-or other-initiation as observed in [138].

Further reference to conversational analysis and turn-takings are in [140, 141, 142]. Third turn repair and beginnings in the telephone conversations are in [143, 144]. A list of references on conversational analysis is in [145]. The ideas expressed in this section will directly and indirectly influence fixing the number of floors. *Prima facie*, conversational analysis has some valuable suggestions to offer about the behaviour of participants and their social organisation in a conversation. Nonetheless, the status, position and other significant social or professional qualifications of a participant also helps in a smooth conduct of a conversation or meeting. With these discussions, it is apparent that a *small* number of floors N is sufficient. Now an argument follows to decide how small is *small*.

[10]The reasons for using the word *Repair* according to [138]: "The term correction is commonly understood to refer to the replacement of an error or mistake by what is correct. The phenomena we are addressing, is however, neither contingent upon error nor limited to replacement. Repair is used rather than correction in order to capture the more general domain of occurrences".

4.6 Solution for Fixing N_{Max}

In a voice-only conference, N_{Max} is the number of audio *channels* referred to also as floors in the context of audio conference. If $N_{Max} = 1$, best speech quality would be achieved. Then any floor control will make the participants too constrained and, in this sense, the conference itself will fail to mimic acceptably closely a face-to-face, real-life conference. Alternatively, if the participants somehow adapt to ensure that there is no more than one speaker at any time, the conversation may be unnatural. In order to arrive at a suitable N_{Max}, the findings of Sacks and Schegloff on properties of conversations, turn-taking (Section 4.5.1) and repair mechanisms (Section 4.5.3) are used in the following discussions. Above observations and many research findings of Sacks and Schegloff [123, 137, 140, 141, 142] on conversational analysis are gainfully used in the sequel.

> **Remark**: Here a formal representation such as propositions are used to arrive at a reasonable solution for fixing N_{Max}. Use of this formal method for expressing the solution is felt necessary though it appears *contrived*. One might suspect the necessity of this formalism since the arguments are based on hitherto non-quantified sociological findings *viz*, conversational analysis. The use of formalism here is only to tighten the presentation of the arguments to the extent possible, and then, work towards a solution.

Now a simple proposition is stated.

Proposition 4.1 *In a voice-only conference, $N_{Max} = 1$ (a) is necessary; (b) is desirable; and (c) is insufficient.*

Proof: *Part (a)* Trivially true.
Part (b): Desirability stems from goodness of speech quality as remarked above. Though utopian, it is indeed desirable that the participants conduct themselves so that no two of them will speak concurrently (Figure 4.2(a), similar to the one in [45]).
Part (c): Investigations into conversational psychology [123, 146] and turn-taking repair mechanisms [138, 137, 139, 136] (Sections, 4.5.1 and 4.5.3) have been reported. Providing for interruptions will render the conference closer to a real-life face-to-face conference. Evidently, an interruption cannot be registered unless there is a provision for at least two simultaneous speech streams (Figure 4.2(b)). ■

> **Remark**: A meaningful model of a conversation must subsume turn-taking mechanism. This is not possible without a feedback and thus the above proposition is consistent with the observation number 4 of Section 4.5.1.

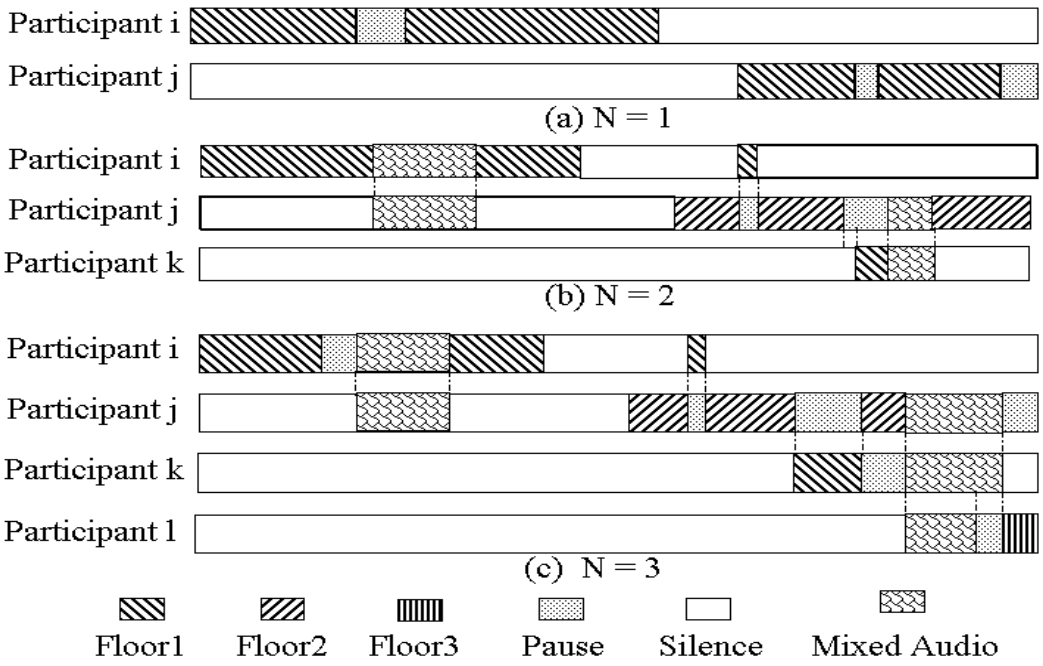


Figure 4.2: **Examples of turn-taking by members in a conference**

4.6.1 Mixing of audio streams

Mixing of audio streams is necessary for a conference. As a prelude to further debating the concerns that dictate the specification of the number of floors, N, it is necessary to take a look at the mechanism of mixing.

With multiple active audio sources, the sound or pressure wave incident on the human ear is a sum of the individual pressure waves [147]. To quote Gyton and Hall [148], "In the case of sound, the interpreted sensation changes approximately in proportion to the cube root of the actual sound intensity". With multiple speakers in the same room the signal captured by a microphone would be a linear combination of the signals. To get this effect with speakers in different locations and generating audio packets, the mixed stream should be the sum of the generated streams. If $X_i(j)$ be the j^{th} linear sample of the i^{th} audio stream, then the j^{th} linear sample of the mixed stream is given by

$$SUM(j) = \sum_{i=1}^{N_{Max}} W_i X_i(j) \tag{4.2}$$

where W_i is the weight or the gain factor for the i^{th} stream. Equation (4.2) forms a basis for a generic mixing algorithm [147]. The amplitude of $SUM(j)$ cannot be increased beyond a certain level (limited by the supply voltage to the sound card of the computer). W_i's are chosen therefore, in interval $[0, 1]$ and sum of W_i's is equal to

one. This avoids clamping. Unbiased or fair mixing demands $W_i = \frac{1}{N_{Max}}, \forall i$.

For large N_{Max}, fair floor sharing is trivial which also has the advantage of less number of participants waiting for floor. On the other hand, if too many participants are concurrently accessing the floor, then it is possible that each weight is so small as to result in deterioration of speech resolution. Thus there is a strong case for specifying an upper bound for N_{Max}. The design must strike a balance between conflicting criteria by fixing the least possible value for N_{Max} with an acceptable quality.

When there are many active members in a conference there can be simultaneous speech streams. A look at some parameters of performance directly implies that a listener can hear, discern and understand only one audio stream and this is the normal situation during any serious, meaningful discussion. *Generally*, human brain cannot register and comprehend more than one audio signal at the same time. A qualitative experiment in support is reported in Section 4.6.2. Psycho-acoustic impact of mixing two or more streams is not considered here. It is argued with an assumption that the conference is "well behaved" and fair (Section 4.5.2). However, the impact of mixing many streams is to convey to all the listeners that there is more than one participant eager to initiate speech. During a deadlock to take over the floor (even in a face-to-face real-life conference) the participants who are trying continuously sense that they are not successful in effectively communicating, and therefore they would try harder to get across the message that "they have a significant point to make". This is enabled if mixing is done.

Even in a conference as one-sided as story telling, there is some feedback to the storyteller and some interactions amongst participants. These interruptions are very short-lived [90]. Thus the idea here is to avoid explicit "Floor Request" (FR) and "Floor Grant" (FG) messages [46, 45] to participants so that they can speak. There can be some provision for participants who do not get a chance to speak while competing with others. It is called a compensatory floor discussed in Section 4.6.6. Explicit FG messages hinder liveliness as it precludes impromptu speech by blocking natural interactions. A large number of floors hamper speech quality even though it allows more liveliness in a conference merely by eliminating FG messages. Therefore there must be more than one floor, but not too many. Now the aim is to find the minimum number of floors. The following definitions are used in further discussions.

Definition 4.2 ***Pause*** *is absence of a participant's voice activity for duration*[11] *of at most* τ.

[11] Generally, a pause can be defined as the region between two utterances, during which the speaker is not articulating. τ is approximately 650ms [149].

Definition 4.3 ***Silence*** *is absence of a participant's voice activity for duration*[12] *greater than* δ ($\delta < \tau$) *which would make other participants feel that the speaker has stopped speaking.*

Definition 4.4 T_+ *is a transition of a participant's state from silence to speech.*

Definition 4.5 T_- *is a transition of a participant's state from speech to silence.*

Proposition 4.2 $N_{Max} = 2$ *is (a) necessary; (b) insufficient.*

Proof: *Part (a)* follows from *Part (c)* of Proposition 4.1.
Part (b): A second token was made available to permit a participant j to interrupt a speaker $i, i \neq j$. It is possible that both i and j are not silent thereafter. This will not pose a problem in a conference of well-mannered participants as either i or j can undergo T_- in turns. Yet, such prolonged and infrequent occurrences cannot be ruled out. Then, the conference would become impolite, messy and becomes a dialogue between two parties in the absence of an intervention by a third participant. Therefore $N_{Max} = 2$ is insufficient. ■

Proposition 4.2 indicates the need for finding a larger N_{Max} that overcomes this limitation. It is but natural to ask at this stage whether *any higher* N_{Max} will suffice. Keeping in the background the discussion on mixing and the aim to mimic a face-to-face real-life conference as closely as possible, there is a need for finding an $N_{Max} > 2$. Before discussing a solution to the question at hand, it is instructive to find out how participants feel when more than one speech streams are mixed.

4.6.2 Qualitative study

In this study tem speech samples five of male and five of female were taken from the DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus [150]. One each of the speech samples from male and female were marked and its transcript was made available to the listeners. From these samples, four 'mixed' samples were formed. The first mixed sample had the mix of the marked speech sample and another sample speech from the lot. The second mixed sample had the mix marked speech and two

[12]This is a *Switching Pause* defined as 'a period of joint silence bounded by the vocalisation of different speakers' and is approximately 860ms [149].

Table 4.1: **Subjective quality of mixed speech files (majority opinion)**

Speech Sample	Marked Speech File Mixed With X other speech files			
	$X = 1$	$X = 2$	$X = 3$	$X = 4$
Male	**C**	**C**	**E**	**E**
Female	**B**	**C**	**E**	**E**

A: With No extra Effort, **B**: With some Effort, **C**: With a lot of Efforts,
D: Very Difficult, **E**: Cannot Recognize

more speech samples from the lot. Similarly, third and fourth mixed speech samples are formed, which contained marked speech plus three and four other speech samples respectively. Totally eight mixed speech samples were generated for male and female speech samples.

First, the marked speech is played out to the listeners. They were given the transcript of the marked speech and even played out many times when requested so that they were able to synchronise their hearing to the marked speech sample. This is felt necessary because, in a running conversation in a conference, listeners would get accustomed to a particular participant's speech if the person is speaking for some time already. The length of speech samples are kept around 8 seconds as recommended by ITU-T P.180 [151]. The speech files in TIMIT database are of shorter duration and two to three files are joined at the silent positions to get a long, smooth speech sample.

The task for the listeners was to identify the level of effort required to discern the marked speech sample from the mixed speech samples. They graded the mixed speech on a scale of five subjective grades. Majority results for ten listeners are given in Table 4.1.

The result shown in Table 4.1 disproves the usual assumption that one can understand even in the presence of prominent cross talks. However, while conferencing using a tool on network, after mixing it is difficult to keep synchronised to the present speaker. For marked male speaker's speech, when mixed with another speaker's speech, majority listeners said it is *with a lot of effort* they could listen to the marked speaker's voice. Some reported that they could hear with 'some efforts'. Thus it is bordering the grades **B** and **C** (Table 4.1). But when mixed with two other speakers' speech,

majority still said **C** but others said it is **D**, thus bordering **C** and **D**. When mixed with three and four other speakers' speech, users could not listen to the marked speaker's speech at all.

When marked female speaker's speech is mixed similarly with other speech streams (all from female speakers), the result showed that intelligibility is a little better compared to the result of previous experiment with male speaker speech. This is observed for first two cases as shown in the table. The last two cases when mixed with three and four speech samples the degradation was so much that listeners could not hear the original speech. Better intelligibility for female speech in first case may have been due to the higher frequency so that the listeners could differentiate it from other speakers' speech.

Analysis of a game (a computer mediated collaborative work) by two groups of participants, each group with five members is given in [119]. Amongst the various measurements tabulated on various aspects of collaboration, the audio part is very important here. The result showed that the total time during which more than five participants spoke simultaneously is negligible. Approximately (using Figure 9.15 of [119]) for only 0.5% of the total time did five participants speak simultaneously and four spoke for around 5% of time. These approximate numbers are computed considering the units of overlapped speech activity and precluding the periods of total silence with which it would be still less. The percentages of simultaneous speech measured by Greenhalgh and others thus indirectly support the remarks made here that simultaneous speakers beyond three is infrequent. Moreover, these percentages are for a collaborative *game* played by two teams with visual players. Thus for teleconferencing with only audio involved, the duration of simultaneous speech reduces further and the result of mixing three streams or more induce similar impression to the listeners that there is an essential requirement for "Repair" (Section 4.5.3). That is, in this case the information provided by the mixed speech is very clear – that the speech of any of the participants is *unclear*![13] The experiment of qualitative analysis has brought out this aspect clearly! Mixing, in fact provides an opportunity to communicate the concerns of listeners to the present speaker. Now to the unfinished business of this chapter – to find the number of floors that is sufficient to convey this message during interruptions.

[13]At times this may look like a *Socrates Paradox*: Socrates is reported to have stated: "One thing I know is that I know nothing".

4.6.3 A conjecture

The propositions showed that 'two' floors are not enough and the qualitative studies implied that more than three would render the mixed speech unintelligible. The main result of these discussions is consolidated as a conjecture[8] here.

> ***Conjecture***: *Three floors are sufficient, (i.e., $N_{Max} = 3$), for a voice-only conference over packet networks.*

Thus N_{Max} is fixed to be *three* and this is used as the basis for the setup in the design and implementations of conferencing solution. With the above qualitative study and arguments in propositions, there can be two alternatives for using these N_{Max} floors.

4.6.4 Self-control by participants

Allowing only three participants (using some technique to resolve which three) to speak simultaneously in a conference, there will always be a question as to how these floors are effectively used. Controlling as well as repairing the conversation would be only by the participants themselves, as set forth in the previous sections of this chapter. It is left to the participants to manage themselves and take turns.

> **Remark**: As seen earlier, the model of a conversation has repair mechanisms built into it. Participants manage to take turns so that the conversation can be back onto the rails in the event of overlapping speech. Since there is a feedback to the participants, quite often, self-repairs would ensue (as discussed in Section 4.5.1). Overlapped conversation will never lead to a rubicon because of the consequences of the model that is considered here (Section 4.5.2).

4.6.5 Imposed control

It is reasonably imperative to impose some etiquette in the rare event of the conference getting out of control as above. The permission for a third participant to undergo T_+ is allowed as discussed. However, existence of three floors is restricted for no longer than a duration Γ. Indeed, dignified speakers stop speaking as soon as speech becomes unbearable [138]. Γ must exceed the pause, τ, so that the present speaker would be able to get the feedback from the third participant who secured the permission to speak.

[8] "How often have I said to you that when you have eliminated the impossible, whatever remains, however improbable, must be the truth?" - Sir Arthur Conan Doyle in *The Sign of Four*

Typically, simultaneous speech would not exceed approximately 400ms [149] – i.e., before one of them initiates repair – in a real-life conference without visual feedback. Therefore Γ needs to be lower bounded by 400ms plus the end-to-end delay. In the worst case it could be about 900ms (taking end-to-end delay including playout delay as 500ms).

Which of the three participants concurrently holding the floors will be forced to undergo $T\_$? An answer to this will be based on Loudness Number to be formulated in Chapter 5. If N is forced from 3 to 2, and it remains 2 thereafter, when will the third floor be made available next? The requirement of a delay Δ (lower bounded by Γ) to allow the third floor after it has been disabled is nothing but the settling time. If it is set high then tight control would be forced because of non-availability of the third floor for the duration Δ. This non-availability of the third floor does not hinder the conference keeping in view that there are still two floors freely available for the participants. Reducing the availability of floors is only to aid faster repair. The measured average vocalisation time for blind conference is around 1680ms [149]. Therefore it would be lower bounded by this value plus the end-to-end delay. Thus it is fixed to be at least 2180ms.

Note that these numbers are debatable and it may vary with the perceptions of the participants and/or implementers since the values for Γ and Δ are dependent on many aspects of conversations and situations including context, language, etc. Choosing $\Delta = 0$ makes this control equivalent to self-control. This imposed control has one advantage that it acts as a catalyst to bring in a faster repair by withholding one floor so that one contender is forced to lose the floor. Implementing this, using the proposed architecture (Section 3.5.2) is very simple as the selectors would have to select up to three speakers in the beginning but it can reduce to two once it finds three speakers, and automatically switch back to three by keeping track of packet periods to account for the settling time.

4.6.6 Compensatory floor

In a large scale conferencing system there can be many participants who may not be able to get into the overlapping speech let alone getting a turn to speak, when there are no tight floor allotment policies. Recognzing this difficulty, the following discussion suggests a scheme to overcome this infirmity.

It is felt [152] that the participants require a floor reserved for this purpose and allocated using some tight floor control methods [44, 45]. This floor is called a compensatory floor. A moderator who is allowed and heard all the time on this floor may

help those who could not get a chance to speak. The 'Loudness Number' discussed in Section 5.6, which is used to select the active speakers dynamically can be always made high for the moderator. This results in moderator being heard whenever (s)he speaks on this compensatory channel. The other three floors are anyway open for competition and interruptions. The compensatory floor is a resource that has to be shared amongst participants but with tight control.

As seen in Chapter 3, a single controller such as Call Processor (CP) can allocate this floor on first-come-first-serve basis. When the CP functionality is also distributed, there is a need for using distributed mutual exclusion algorithms [153, 154, 155, 156, 157]. These algorithms work on the total set of participants registered with many control units (CPs).

Till now it is assumed that a fourth, a separate channel was used for the compensatory floor. Another suggestion is to make one of the three floors a compensatory floor since three floors are sufficient. The floor which is earmarked as 'compensatory floor' may also be used in general competitive quota in the absence of a request from moderator or any other user and it can be withdrawn when requested. The decision whether N_{Max} needs to be three or four, inclusive of compensatory floor, is left to the application developer who has to take into account network capacity, etc. On the test-bed developed as part of this thesis, N_{Max} can be set between two and four.

4.7 Limitations

Though there has been much discussion within Computer Human Interaction (CHI) community about using the social aspects based on ethnomethodological inferences, there are some voices that question the much-hyped use of it. Two main themes in ethnomethodological work are vastly considered – (a) the organisation of social action and interaction and the organisation of work, and (b) how these have been applied by those who claim an ethnomethodological affiliation to the study of social action, interaction and work involving technology. Quoting Graham Button [158], "Taken together, ethnomethodologically affiliated studies have produced a strong critique of the design of technology at work for they have displayed that technology, at best, often fails to support the work it is designed for, or at worst, does not allow people to actually engage in their work, because the technology is not aligned to the practices through which they organise their actions, interactions and work".

Heath et al. [159] observe that "there are relatively few examples of successful applications in real world settings. It is suggested that the lack of success of CSCW systems

derives not so much from their technological limitations, but more from their insensitivity to the organisation of work and communication in real work environments".

Graham Button describes two paradoxes [158]; (1) The technology designed to support 'large scale' activities transforming the 'small scale' detail of action systematically undermine exactly the detailed features of work practices of large scale activity. (2) The unavoidable transformational nature of technology and system design in work settings may make ethnomethodology powerless as its tradition is in analysing the practice and not in 'inventing the future'. These discussions show some conflicts on how best design may learn from ethnomethodology.

In a face-to-face conversation, listeners hear the speaker immediately as they are speaking. The incremental nature of this production, turn-so-far by turn-so-far, word-by-word, syllable-by-syllable, phoneme-by-phoneme, means that a listener hears each component sound at the unfolding cusp of the time as it is produced. It is this facility that allows somebody to interrupt at a key moment to finish off an utterance, as it enables them to analyse what they have been hearing bit by bit, and jump in appropriately. This process must work effectively in the production of synchronised utterances. But being on the network, introduction of end-to-end delay would pose some problems. They are typically: (a) unintended interruptions due to lack of perceived response; (b) rephrasing due to expectation of a dis-preferred response; and (c) mistimed feedback [160]. There are some techniques in [160] about how participants can adjust to the delay that is inherent in these CSCW work environments. It is known by experience that users gradually adapt to the effect of delays [161]. Long distance satellite phone calls are a case in point. The effect of delay is not explicitly taken into account here in the discussion. It is presumed that the application would strive to minimise end-to-end delay. Nonetheless delay merits attention for any realtime application.

The other aspect that was not considered here in the discussion (though related to the turn-taking) is the underlying network usability. A problematic network support may lead to difficulty in turn-taking. Sometimes, silences may be wrongly interpreted or attributed to technical faults [133]. A good discussion on the problems that arise, and some suggestions for the network support including the tool used for CSCW are in [162].

4.8 Conclusions

This chapter stipulates a bound on the number of floors through insights from ethnomethodology and some qualitative analysis. The number of floors fixed seems reasonable for a conference. This small number would avoid unnecessary bandwidth usage in the context of the architecture proposed earlier (Chapter 3).

It is recognised at this moment that there may be more insights when large-scale deployment of conferencing happens. It is expected some more interesting observations, even surprises and challenges are in the offing with large-scale deployment of conferencing. Philosophically, it is the *end of the beginning*[3] of building an application that can provide the sensation of a real face-to-face interaction with the use of technology. There will be many surprises and an equal number, if not more – of iterations in application development as and when large-scale conferences throw up surprises. Thus there is no pretence of having spelt out the last word on the number of floors. Nonetheless, the strong feeling is that the number will not change significantly, on second thoughts, will never ever!

4.9 Retrospection

At the end of these discussions, more often than not, it is felt that a mild inner voice of dissent is yet unsilenced. Observations are, no doubt, reasonable in themselves. How far are these observations to be taken seriously? Others might query such forceful use of conversational analysis when the qualitative study showed that it is practically futile to mix more than three speech streams. The considered reply here is that conversational analysis alone can be used to set the upper bound on number of floors (N_{Max}).

Conversational analysis is a basic and a very well established work by Sacks et al. [123] three decades ago. Several researchers working in the field of CSCW and CHI use this branch of ethnomethodology to different extents to tackle many design issues for human interface with computers. Thus it is always advisable to have insights purely from the users and sociological perspective before endeavouring the design of solutions. Indeed, there is no denying that the need to converse is basically sociological, – technical or otherwise. With so much of methodical literature on behaviour and dynamics of a conversation, it is worthwhile to have an in-depth knowledge that helps in CSCW application design. In a face-to-face real-life conference because of the visual clues

[3]Now this is not the end. It is not even the beginning of the end. but it is, perhaps, the end of the beginning - Sir Winston Churchill

and direction of speech perception, the participants are able to find and lock-in to the speakers of their choice almost always. However, in a blind conference (with no visual clues) it hardly matters whether it is three participants or higher number of them speaking. As noted previously, in the beginning of this chapter (Section 4.1), *functional* aspects are important and must be taken into account before designing the application. It is also clearly recognised here that many a time it is difficult to segregate *functional* aspects from the *form*. Without these discussions and insights one can fix the number of floors. However, more often, in trying to provide a real life service, there may be unnecessarily a large number of floors allowed (as seen in [27, 38, 39]). The very approach taken in this thesis is therefore believed to be novel for the problem that is solved here.

Chapter 5

Loudness Number

5.1 Introduction

As discussed in the previous chapter, there are many related studies with respect to conference controllers [45, 47, 130] that are used to share a medium or a resource in a computer supported collaborative work (CSCW) using Floor Control. Many studies on floor control are reported by Hans Peter Dommel in [44, 45, 46]. All the studies deliberate on effective usage of the resources in a simplistic manner with a stress on fairness. However, for a voice conference, sharing the resource (a voice channel), with explicit tokens and messages using a centralized or distributed controller working on the principle of mutual exclusion [47] makes the conference too artificial. This is true even when there are many floors to offer because of the transactions that take place before granting a floor. In such a case, if there are many participants waiting for a token to speak, a participant cannot interrupt and speak immediately to the current speaker. This type of conference implementation would render the conference very tight, unlively and inflexible. A shared voice medium will never take multiple speakers at the same time and cannot be shared like other resources, such as a white board. A shared voice medium is very dynamic, vibrant, and self-contained, and depends on the socio-psychological involvement of participants. With fixed N_{Max} number of floors, the problem as to how to allocate these floors dynamically to the contending participants is discussed in this chapter.

In a real-life face-to-face conference there is always a possibility that a highly vocal[1] speaker gets the attention and the participants are permitted to interrupt a current speaker anytime. This implies that anybody can just speak to the audience without obtaining prior permission or providing notice in advance. There may, however, be other reasons for not interrupting, when, for example, a dignitary is addressing.

[1]The sounder your argument, the more satisfaction you get out of it. - Edward W. Howe

The utility of any application that provides a conference facility would be enhanced if it is rendered as close as possible to real-life conference. This chapter investigates the issue of making a voice-only conference acceptably close to real-life face-to-face conference. For this, adaptive speaker selection on the basis of a figure of merit of a speaker is proposed. With this quantifier, speakers are ranked and selected, making this conference a "controller-less and automatic floor controlled" conference. Indeed, this chapter is in a way dependent on ideas expressed in Section 4.5. The metric defined in this chapter is implicitly used for floor control discussed in Chapter 4.

5.2 Speech

It is worthwhile to have an appraisal of Speech in respect of its origin and characteristics as a precursor to the discussion on the new metric to be called *Loudness Number*. This study of speech would facilitate Voice Activity Detection (VAD) to be investigated in Chapter 6.

The vocal tract produces speech. Speech signal in the form of pressure waves - varying with respect to time - travel in a material medium and fall the diaphragm of the listener's ears. It is non-stationary, changing when the vocal tract vibrates. Speech is divided into sound segments that share some common acoustic and articulatory properties with one another, though for a short duration. Vocal tract articulators are vocal folds, tongue, lips, teeth, velum and jaw [163, 164, 165]. There are voluntary vocal tract movements and specific positioning of vocal tract articulators.

5.2.1 Speech and its characteristics

Speech has two main building blocks – vowels and consonants. While vowels allow unrestricted airflow, consonants restrict airflow and have lower intensity. Almost all speech is produced during exhalation. Speech production may be viewed as an output of the vocal tract as a filter, excited by a sound source. The source may be periodic or aperiodic, resulting in voiced speech or unvoiced speech respectively. The voicing source is in the larynx at the base of the vocal tract where airflow is interrupted periodically by vibrating vocal folds. The pulses of air produced by the abduction and adduction of folds generate periodic excitation for the vocal tract. However, a small deviation from this frequency is inevitable as the vocal tract is non-linear. This is in fact the nature of human speech that renders it difficult to synthesise [166]. Unvoiced speech is noisy due to random nature of the signal generated at a narrow constriction in the vocal tract.

For both voiced and unvoiced speech the vocal tract vibrates at frequencies depending on its size and shape. Energy in those frequencies is reinforced due to resonance. Energy in other frequencies dissipates rapidly. Resonant frequencies are seen as peaks in the spectrum and are called "formants" [164, 167, 168].

As a periodic signal, voiced speech spectrum consists of harmonics of a fundamental frequency, $F0$. The spectrum is not purely discrete in multiples of $F0$ as the vocal tract changes shape continuously. Speech signal is rarely periodic for more than 40ms because of movements of vocal tract articulators. Hence speech is often called quasi-stationary.

In further discussions here, the word 'speech' refers to spoken English unless otherwise specified. Some parts may hold relevance to other languages also. Periodicity, duration, boundaries of sound segments, amplitude relationships in sound sequences are all obtained from recorded speech. The important characteristics of speech signal as relevant to this study are: (a) amplitude and time-frequency behaviour in a sound stream; (b) quasi periodicity of voiced speech during segments of utterances; and (c) features of gradually changing patterns between different sound segments. These characteristics of speech are seen in all languages with some variations. Speech signals differ in different languages normally in the place of articulation.

5.2.2 Phonemes and syllables

Each language has a minimal set of units of speech sounds called phonemes which serve to distinguish one word from another. Thus a phoneme is the smallest contrastive unit in the phonology of a language. Sounds associated with some phonemes frequently have some articulatory gestures or configurations in common. Each word is a series of phonemes corresponding to the vocal tract movements needed to produce the word. Words are typically divided into phonological units called syllables. Each syllable has one vowel that causes its most intense sound.

Languages share phonemes such as cardinal vowels but differ slightly for these sounds. English words differ from each other in some aspect of their phonemic composition. English provides good examples of sounds used in various languages [163], which change mostly in the place of articulation. English is popularly considered typical of most non-oriental languages in voicing sounds and has about 40 phonemes.

Terminology

Apart from vowels and consonants there are a few more speech sound types [163]. (1) Diphthongs: They are almost similar to vowels in which the tongue and lips move between two vowel positions. E.g.: 'i' in bite. (2) Glides: Similar to high vowels but

employ narrow vocal tract constrictions. E.g.: First 'w' in wow. (3) Liquids: Similar to vowels but they use the tongue as an obstruction in the oral tract, causing air to deflect around the tongue. E.g.: 'l' in lull. (4) Nasals: In nasals the air flows through nostrils. Depending on the language some vowels or consonants can be nasals. E.g.: 'n' in none. (5) Plosives: Consonants involve the complete closure and subsequent release of vocal tract obstruction. E.g.: 'p' in pop. (6) Fricatives: Sounds that employ a narrow constriction in the oral tract. E.g.: 'sh' in shoe.

5.3 Motivation for Defining a New Metric (Quantifier)

Speech contains sentences formed by groups of words. Each word is made up of phonemes and syllables. Hearing is the sensory perception of the sum total of different air pressures caused by these phonemes. Typically, speech consists of many words and/or sentences. To identify whether a person is speaking or not, it is necessary that there is some sound signal caused by the airwave pressure on a microphone. It is further desirable to find if the audio signal is present over a protracted period as in speech or is a sudden spike as in a clap or noise burst.

As already explained in the Section 4.6, there cannot be more than N_{Max} speakers speaking in a conference at any time. Thus the problem frequently faced is to select a set of a few (N_{Max}) persons who are allowed to speak out of all the participants participating in the conference [169]. In the context of video conference, for instance, it has been remarked [170]: "...it was not obvious how to determine which sounds from the audience were appropriate to transmit". For a voice-only conference it is a more difficult proposition to tackle in the absence of visual cues.

In this backdrop, the questions faced are: (a) Whose voice is to be selected? and (b) What is a criterion to make such a selection? For humans it may be natural and easy to decide as a person decides on the basis of the importance[2] of the speaker and the matter spoken. It is based on an individual's judgement. Also, speech may be interesting or otherwise. For a computer to take a decision, there must be a figure of merit to compare and decide. These issues faced by a computer in the context of non-mediated conference are focussed upon in the sequel.

[2]Like an immediate superior! or a hated subordinate!

5.4 The Problem of Selecting N_{Max} Speakers

The problem to be solved by selectors is as follows. For each mixing interval (packet period), how should it choose[3] N_{Max} (see Sections 4.6.3, 2.4.3 and also Chapter 3) packets out of M it might receive. One way to do this would be to allow a speaker who has highest energy at that instant. That is to rank the M packets received according to their energies [89], and choose the top N_{Max}. This is usually found to be inadequate because of following reasons.

A. Random fluctuations in packet energies can lead to poor audio quality (as mentioned above, the duration of vocal tract articulators being in a particular position for not more than 40ms). For example, if a participant reaches a high transient while speaking or if background noise burst were to be incident, then this packet may be chosen amongst N_{Max}. This is precisely the reason which makes the Audio Processing Unit (APU) (Section 2.4.3 and Section 5.7.3) not suitable.

B. Speech may be voiced or unvoiced. A fricative like *'sh'* in *sheep* is unvoiced. Its packet energy is found to be far less than that of the next packet that contains the last syllable *'p'* in *sheep*.

Therefore consideration of merely one packet time for decision is flawed. Thus there is a need for a more comprehensive metric different from mere individual packet energy. The metric required should have the following desirable characteristics:

A. A person who is speaking (i.e., has the floor) should not be *easily* cut off by high energy transient spikes from other participants. This implies that current speakers[4] should have some weighing factor depending on their past activity; this weight is often referred to as *Persistence* or *Hangover*. Indeed, relevance of a speech stream can be assessed, mostly, after listening to at least one complete sentence.

B. A person should not be allowed to interrupt others who are speaking simply because of a high-energy packet a moment before and for a short duration. Hence a participant who wants to interrupt the current speaker will have

[3] We can try to avoid making choices by doing nothing, but even that is a decision.

- Gary Collins

[4] The trouble with her is that she lacks the power of conversation but not the power of speech.

- George Bernard Shaw.

to raise the voice and persist for a little while in order to break in or to be heard, that is, to get into the set S.

5.5 Discussion on Requirements

In a real-life conference, a slight gesture of a participant or some small sound often indicates a desire to interrupt. However, in voice-only conference, a participant's intention to interrupt can be conveyed only through a speech-based quantifier. On this basis the participants are selected as one of the N_{Max}.

S is the set of participants in conferring mode i.e., participants who got selected as one of N_{Max}. Clearly, N_{Max} is the cardinality ($|S|$) of S. In an ongoing audio conference a person who is presently speaking must be accorded some priority over others who are not speaking. Therefore a participant presently not in S but wants to get into the conferring mode should have a suitable mechanism to do so. This causes one of the participants in the set to make way as $|S|$ is fixed. This has to be done in such a manner that the present speaker should not be removed from the conference abruptly even when that person is not speaking for a short duration. Also, those who want to get into S should be able to do so but with a nontrivial and modest effort, yet smoothly.

The above discussions show that the proposed technique is slightly biased towards the current speaker. To a certain extent it is so and without this bias there will always be an abrupt end of speech of a current speaker[5]. As a matter of fact, in a well-behaved conference, repairs (Section 4.5.3) take place and consequently after repair, only one person starts addressing the participants. This leads to other floors in N_{Max} being vacant. Therefore it is easy to get access to a floor. The new speaker would get into S at the cost of a speaker who is not contributing to the conference at present as well as for some period of time in the past. This main requirement of close-to-seamless change in the composition of the set S is addressed here.

With the above requirements, a speech metric has to be defined that truly represents the conferencing situation for allowing a new participant to the conferring mode. Now to define such a quantifier, the concerns that arise are:

- Features of speech that should be considered; and
- Actual extraction of these features from a participant's speech.

[5] "Would ye both eat your cake and have your cake?" - John Heywood.

Tracking all the parameters of speech (amplitude, pitch, etc., Section 5.2.1) leads to massive computation and confusion as many of the parameters change fast and almost on a per packet basis. Clearly the decision must consider the speech activity of a participant for at least a few seconds. For this, *amplitude*, which is a primary characteristic of speech cannot be neglected, the reason being that amplitude must be higher than the ambient noise if a person is speaking. To reduce the computation requirement for a realtime application, only the amplitude is considered hereafter.

The metric should be able to take into consideration a participant's speech amplitude level at present (immediate past), the history of speech amplitude level for a well defined shifted past, and speech activity that gives information about fraction of the time the participant was speaking over a longer past duration. This dynamic metric should change smoothly with time as the speaker activity changes so that the selection – addition and deletion – of participants for the set S is smooth. This metric is defined as *Loudness Number*[6] (λ), and it characterises a participant in a conference. Loudness number then ranks a participant continuously from the time of entry of that participant.

5.6 Loudness Number (λ)

Loudness number (λ) is calculated for each packet of a participant so that the decision can be taken at the lowest possible measurable at the application layer. Loudness number is calculated and updated on a per packet basis, which corresponds to the least time the conferencing application recognises for mixing different audio streams. The basic parameter used in this calculation is packet amplitude, which is calculated as root mean square (rms) of the audio samples in the packet and denoted by X_K for K^{th} packet, as,

$$X_K = \sqrt{\frac{1}{T}\sum_{i=1}^{T} x_i^2}$$

where x_i is the linearly quantised amplitude of a speech sample and T is duration of packet in number of samples. Though some hardwares provide $A/\mu - Law$ coding in addition to the linear quantisation, only linear quantisation is considered here because all sound blaster cards used in PCs have linear 8/16 bits coding. Any other coding can be accomplished using linear quantisation. The energy of a packet may also be used instead of amplitude but variation of energy is more compared to the rms amplitude

[6] "Number rules the universe" - The Pythagoreans

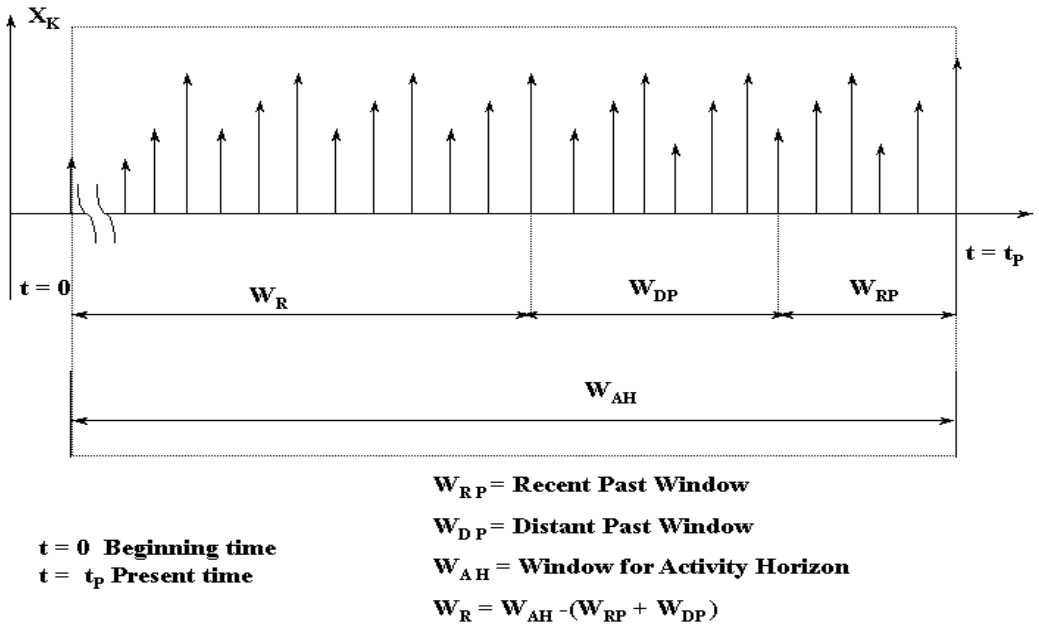


Figure 5.1: **Illustration of various windows for λ calculation**

of samples. Three windows are defined as in Figure 5.1, for calculating λ. These three windows contribute to the three parameters that are to be considered while calculating λ.

A. The present amplitude level of the speaker is found by calculating the moving window average of packet amplitude (X_K) within a window W_{RP}, called "Recent-Past Window" starting from present instant to some time in past.

B. The past amplitude in addition to the present amplitude of the speaker is found by calculating the moving window average of the packet amplitude (X_K) within a window W_{DP}, called "Distant-Past Window" which ends at the point where the "Recent-Past Window" starts and stretches back into the past for a pre-defined interval.

C. The activity of the speaker in past is found with a window W_{AH}, called "Activity Horizon" spanning recent-past window, distant-past window and beyond. Clearly, $W_{AH} > W_{RP} + W_{DP}$.

Though the contribution of the activity horizon looks similar to the contribution of the recent-past and distant-past windows, activity in the past is calculated differently from that in horizon window.

The quantities during these three intervals are defined as L_1, L_2 and L_3 respectively. L_1 quantifies the recent past speech activity, L_2 the distant past speech activity and L_3 the speech activity in the activity horizon window. L_3 yields a quantity that is proportional to the fraction of packets having energies above a pre-defined threshold. The threshold is same for all the participants to enable a fair comparison. The operating volume level for each participant can be set to the same value with the help of front-end application program or by incorporating an automatic gain controller. This is a necessary condition for comparing the λ of all participants before selection as λ depends on threshold θ.

5.6.1 Definition of loudness number (λ)

Loudness Number λ_{t_p}, at the discrete time instant t_p, measured in packet time units, is defined as a convex sum of L_1, L_2 and L_3 with weighting factors summing to 1. The definitions for L_1, L_2 and L_3 are shown below.

$$L_1 = \frac{1}{W_{RP}} \sum_{K=t_p}^{t_p - W_{RP}+1} X_K \tag{5.1}$$

$$L_2 = \frac{1}{W_{DP}} \sum_{K=t_p-W_{RP}}^{t_p - W_{RP} - W_{DP}+1} X_K \tag{5.2}$$

$$L_3 = \frac{1}{W_{AH}} \sum_{K=t_p}^{t_p - W_{AH}+1} \theta I_{\{X_K > \theta\}} \tag{5.3}$$

$$\text{where, } I_{\{X_K>\theta\}} = \begin{cases} 1, & \text{if } X_K > \theta, \\ 0, & \text{else.} \end{cases}$$

and θ, $0 < \theta < \max\{X_K\}$ a threshold. The average over window is taken with time running backwards to signify that the previous X_K are considered (Figure 5.1). Here the windows are rectangular. For experiments, θ is chosen between 5% and 20% of the maximum amplitude of the voice samples of a packet for 8 bits quantisation. For 16 bits quantisation θ is about 20% of average packet amplitude. The θ values are not identical for 8 bits and 16 bits sampling because of the obvious difference in quantisation for the same signal. For example, if the speech signal is limited to $[0, 5V]$ band, then a sample value of $2V$ in 8 bits quantisation yields a value of 102 where as 16 bits quantisation yields 26214. A heuristic procedure is given in Section 5.8.1 to find out a typical value for θ.

Now, the Loudness Number, λ_{t_p}, at t_p or for the present packet is calculated as

$$\lambda_{t_p} = \alpha_1 L_1 + \alpha_2 L_2 + \alpha_3 L_3 \tag{5.4}$$

$$\text{where } 0 < \alpha_1, \alpha_2, \alpha_3 < 1 \text{ and } \alpha_1 + \alpha_2 + \alpha_3 = 1$$

λ_{t_p} varies slowly because of the moving averaging used. By choosing appropriate values for windows, α_1, α_2, α_3 and θ, dynamics of λ can be adjusted so that a participant can be added or deleted from a conference smoothly and seamlessly.

5.6.2 Discussion on loudness number

It may be seen that all the requirements listed in Section 5.4 are reflected in the definition of loudness number. λ tracks the input with some delay depending on the size of the windows. Rectangular windows imply that the weight for each X_K in its domain is the same. Also, as the influence of the packet energies taken over a small span of samples is to be same, the use of standard windows with smooth variations – for example, Hamming or Hanning windows [171, 172] – is ruled out. Moreover, different windows enable different amounts of influence (using α's) with respect to present and past voice activity of a speaker. For example, by changing the window length W_{RP} and α_1 to a higher value λ may be made to depend more on present speech activity. The contribution of L_3 to λ is different as it is a fraction of the time a speaker was active with energy above θ. Even if a person is speaking with less volume but uninterruptedly then (s)he is expected to have a higher λ compared to the one who speaks less frequently with long pauses. Hence a speaker who tries persistently will get a chance to get into the set S. Also, a speaker who is in set S for substantial duration is not removed from S easily.

Variations of λ is given in Figure 5.2(b) with Recent-Past window, Distant-Past window and the Activity Horizon chosen as 5 seconds, 10 seconds and 30 seconds respectively and α_1, α_2 and α_3 are 0.4, 0.3 and 0.3 respectively. The value of $\theta = 200$ (for 16 bits quantisation) is used. These numbers are arrived at after many trial runs on the audio conferencing *Proof of Concept Setup* (PoCS) and through a limited survey and feedback from users. Lengths of the windows are directly coupled to the users' perception of duration of *time* the other participant was speaking. Thus these numbers may vary widely with each pair of individual listener and speaker. The numbers above represent the average and may vary with language and even the topic of discussion. Exhaustive results on its variability are not available now as it takes enormous resources and manpower to obtain them. λ varies with X_K (Figure 5.2(a)) slowly, depending on the length of the windows and values of each α. On the PoCS for testing there is a provision to select window sizes and α's (Figure A.4). Before probing into λ further, alternative proposals are presented here.

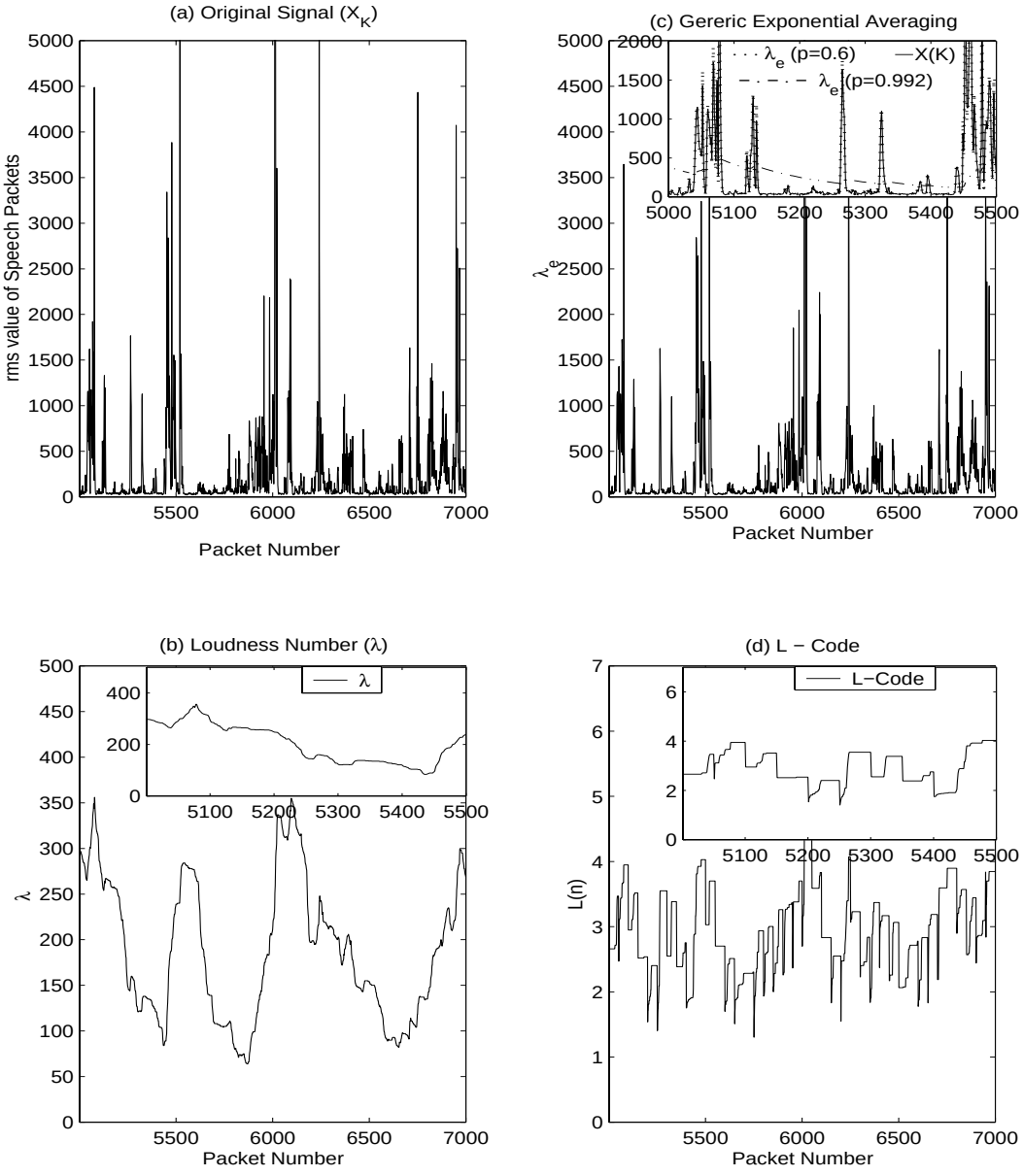


Figure 5.2: **Illustration of λ, λ_e and *L-Code***

5.7 Related Studies

The generic exponential averaging and a proprietary algorithm are presented in this section. Both these methods have less computation burden and are easy to implement. However, they fall short of the requirements of a quantifier as listed in Section 5.5.

5.7.1 Generic exponential averaging

A simple alternative with less computation is Generic Exponential Averaging. Loudness number λ_e at any instant K is given by,

$$\lambda_e(K) = p\lambda_e(K-1) + (1-p)X(K), \quad 0 < p < 1 \tag{5.5}$$

The transfer function for this averaging scheme is given by

$$H(z) = \frac{\lambda_e(z)}{X(z)} = \frac{(1-p)}{1-pz^{-1}} \tag{5.6}$$

Here, X_K is used in place of $X(K)$ for ease in representation. It can be easily seen that the weight given to X_K decreases exponentially, as p^n is the weight for the sample $X(K-n)$. Figure 5.2(c) shows $\lambda_e(K)$ for $p = 0.6$ and 0.992. The inset has a close-up view for comparing λ_e with $X(K)$. It is seen that λ_e tracks $X(K)$ very closely for lower values of p. For higher values of p averaging is significant over many past samples and hence is sluggish.

In principle, one can choose p that offers an approximate equivalent window size for three different windows. The advantage of this method is its simple computation. However, computation of λ is also very simple and will be discussed later in Section 5.8.3. The main disadvantage in using this averaging method is that over the window under consideration the weights given to $X(K)$'s are not the same compared to the way λ is computed. Thus λ is preferred to λ_e for experiments on test-bed since the idea here is to consider the past activity as a whole with uniform weight.

5.7.2 'L-Code' algorithm

In some hardware conference bridges speakers are selected for each speech sample by calculating a parameter called "L-Code". One such proprietary algorithm [173] used by Wipro is briefly given below.

Algorithm 5.1 ***L-Code***

Step 1: Initialise DecayCounter = C;

Step 2: For each sample, find Y(n) = abs(log(X(n)));

Step 3: $C = C - 1$;

Step 4: *if*(Y(n) > L(n)) *then* L(n+1) = L(n)/2 + Y(n);
elseif(Y(n) = L(n)/2) *then* L(n+1) = L(n);
elseif((Y(n) < L(n)/2) && (DecayCounter $\neq$0))
then L(n+1) = L(n);
elseif((Y(n) < L(n)) && (DecayCounter = 0))
then L(n+1) = L(n) − 1;

Step 5: *if*(DecayCounter = 0) *then* Set DecayCounter = C;

Step 6: GO to Step 2 for the next sample.

abs(·) implies absolute value. The new L-Code for each timeslot can then be stored in a structure "L-Codes". L-Code update process allows the L-Codes to increase every time if the "loudness" is increasing but only to decrease in value every C^{th} (C is the count for DecayCounter) sample if the "loudness" is falling. This makes it easy for a participant to become an active speaker but "hard" for a participant to stop being an active speaker because, reduction in L-Code value is not proportional to reduction in the amplitude of the speech samples but merely dependent on the information that the amplitude is low. One major disadvantage of this is that a sudden spike of sound from a participant may make that person active though unintended. In turn this may unfairly drop an existing active participant from the conferring mode.

Figure 5.2(d) shows the L-Code values calculated at packet levels instead of on per sample basis because applications at servers can work only at packet intervals. The inset shows a close-up of L-Code values wherein the flat region implies that the L-Code is constant during the downward swing of the loudness till the counter becomes zero. Even if log of the sample is not taken, the trend (of attaining a plateau) remains the same but with higher values for L-Code. The algorithm works well in systems where new L-Code is computed for each sample since each sample is spaced $125\mu s$ apart for speech. For packet intervals of 10ms, L-Codes are found to be inefficient through the pilot runs on the PoCS because of the flat regions in Figure 5.2(d). Certainly, λ in Section 5.6 is more suitable when the decision to add or delete a participant from S is to be made between packets and not at every sample.

5.7.3 Audio Processing Unit (APU)

A module called APU that finds a maximum of four speakers proposed by Kyeong-yeol, et al. [89] and is almost similar to the above method. They base their selection on

the sample amplitudes directly. They rank the speakers based solely on instant energy, comparing the signals of those who are speaking (by finding the speaker activity). Consequently the sample that has more energy ends up being selected though it may be a transient or a spike. To avoid the frequent changes in the set of selected speakers, refreshing of the set is done at 1-2 seconds interval. Their algorithm is implemented in hardware for a centralized conferencing and thus is not scalable. It does not meet the characteristics required as in Section 5.4. For distributed conferencing it is cumbersome because of the requirement of synchronising these units to make the decision. Utility of this scheme with respect to conferencing architectures has been addressed in Section 2.4.3.

5.8 Analysis of λ

Characteristics of λ can be analysed with a few assumptions. For ease of discussion the following variables are defined.

Nonoverlapping Activity Horizon window: $W_R = W_{AH} - (W_{RP} + W_{DP})$ (See Figure 5.1) (5.7)

$$\text{The average over the window } W_{RP}\text{: } \bar{X}_K^{(RP)} = L_1 \tag{5.8}$$

$$\text{The average over the window } W_{DP}\text{: } \bar{X}_K^{(DP)} = L_2 \tag{5.9}$$

$$\text{The average over the window } W_{AH}\text{: } \bar{X}_K^{(AH)} = \frac{1}{W_{AH}} \sum_{K=t_p}^{t_p - W_{AH}+1} X_K \tag{5.10}$$

$$\text{and, the fraction due to Indicator function (Equation (5.3))}\text{: } f_\theta = \frac{L_3}{\theta} \tag{5.11}$$

It is necessary to look at the characteristics of f_θ and θf_θ before finding the properties of λ. For a given distribution of X_K, f_θ varies with θ and takes values in $[0, 1]$. f_θ is inversely proportional to θ. Since θ is defined to be in the open interval $(0, \max\{X_K\})$ (Equation (5.3)),

$$\lim_{\theta \to 0} f_\theta = 1 \text{ and, } \lim_{\theta \to \max\{X_K\}} f_\theta = 0$$

5.8.1 Selection of θ

$L_3 = \theta f_\theta$. f_θ, which influences λ depends on θ and X_K. The peak of θf_θ lies in $[0, \theta]$ for θ in $(0, X_K)$ because of the inverse relation between them. Now the question is, how to choose θ for a given distribution of X_K. Assuming uniform distribution of X_K, θf_θ vs. θ (Figure 5.3(a)) appears similar to the entropy curve of information theory.

By using θ corresponding to its maximum, the change in θf_θ can be maximised, as θ brings sign change in the slope of θf_θ. Now for fixing θ it is recommended to find the approximate distribution for X_K, though X_K in the domain of speech signals is not stationary. Therefore this exercise provides a direction in which θf_θ is skewed, and θ corresponding to the peak is that which maximises L_3 for active speakers.

Effectively, L_3 is used to distinguish speakers who are active from those who were not for some time in the immediate past (Section 5.6.2). Thus by setting θ possibly at a threshold of speech and background noise amplitude, L_3 can be maximised for those who are speaking compared to those who are silent. The distribution (a normalised histogram) of X_K is shown in Figure 5.3(b) for a selected speaker for about 15s in conversation. Thus this sample can be representative of speech and silent packets that are almost likely in a conversation. The cumulative distribution of X_K is shown in Figure 5.3(c). The *cdf* implies that for quite a good part of time a speaker is silent or pausing. Usually, this is observed in a conference. An easily discernable deviation occurs when the active participant is not speaking but is singing a continuous tune or at the other extreme, silent throughout. Therefore the sample considered here has packets from a speaker who is in a conversation such that it includes both speech and silence. Using this *cdf* of f_θ for normalised values of $\theta \in (0, \max\{X_K\})$, θf_θ is plotted in Figure 5.3(d) for normalised values of θ. Clearly, θ lies within $5\% - 20\%$ of X_K. A higher value of θ enables λ to be used for silence suppression as well. Figure 5.3(d) shows only the trend. This is a heuristic procedure based on the measured $cdf(X_K)$. Possibly a better estimate for θ can be found essentially by having more trial runs of the conference tool on PoCS.

5.8.2 Bounds on λ

The characteristics of λ are found using some of the bounds depending on the operating range of X_K. The following are some of the properties of λ.

Property 5.1 $\forall K \in [t_p, t_p - W_{AH}]$, *if* $X_K \leq \theta$ *then* $\lambda_{t_p} < \theta$

Proof: $X_K \leq \theta$ implies

$$\begin{aligned} L_1 &= \bar{X}_K^{(RP)} \leq \theta \\ \text{and, } L_2 &= \bar{X}_K^{(DP)} \leq \theta \text{ (using equation (5.8) \& 5.9)} \\ L_3 &= \theta \frac{1}{W_{AH}} \sum_{K=t_p}^{t_p - W_{AH}} I_{\{X_K > \theta\}} \\ &= 0 \\ \text{Therefore } \lambda_{t_p} &< \theta \text{ (using equation (5.4))} \end{aligned}$$

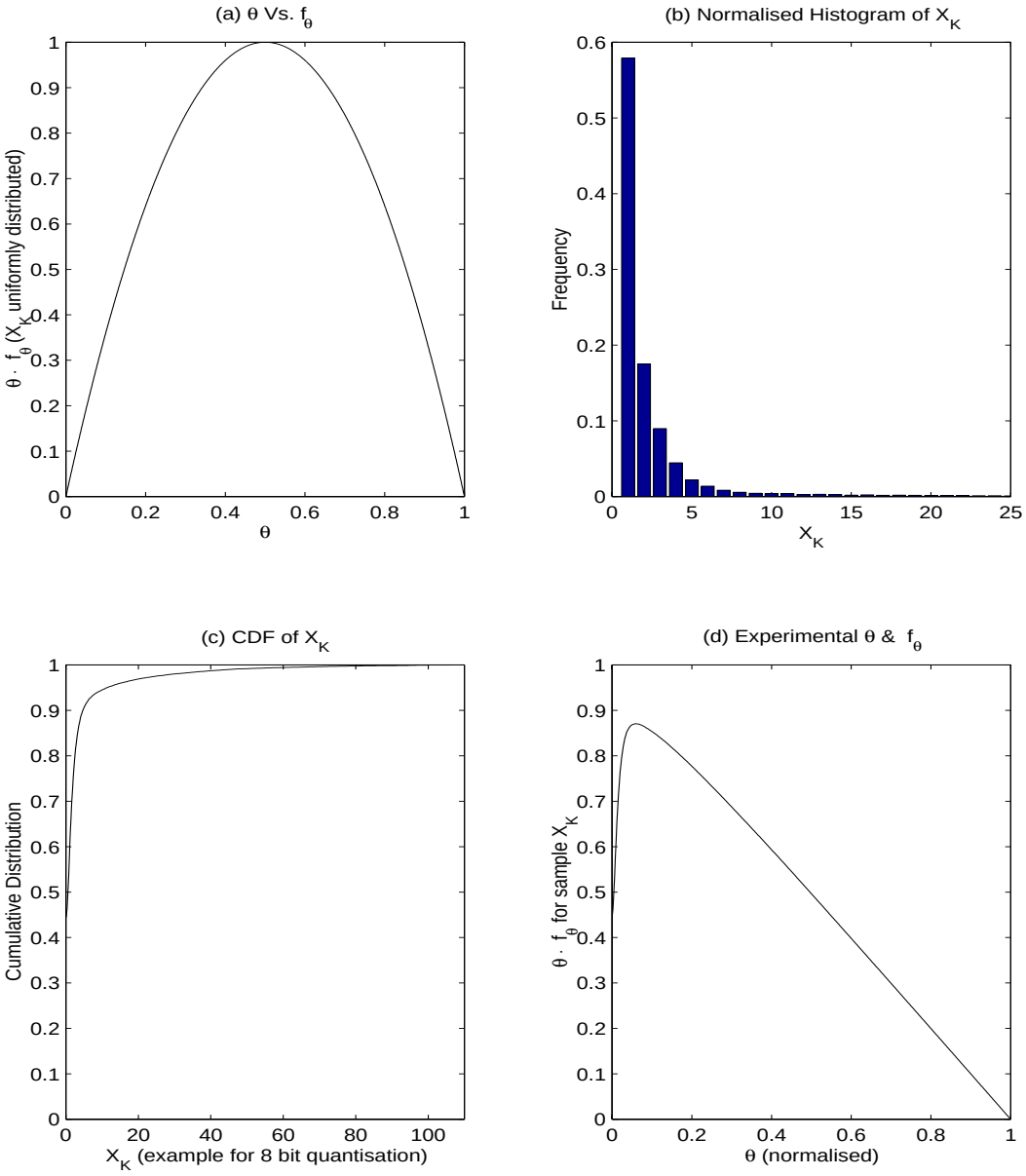


Figure 5.3: **Normalised histogram, CDF of X_K and θf_θ $vs.$ θ**

This expression shows the dependency of λ_{t_p} on θ and it tends to be around the average $\bar{X}_K$ and is actually upper bounded by θ when the speaker is not active in the duration W_{AH}.

Property 5.2 *$\forall K \in [t_p, t_p - W_{AH}]$, if $X_K \geq \theta$, then $\lambda_{t_p} < \max\{\bar{X}_K^{(RP)}, \bar{X}_K^{(DP)}\}$*

Proof: $X_K \geq \theta$ implies

$$\begin{aligned} L_1 &= \bar{X}_K^{(RP)} \geq \theta \\ \text{and } L_2 &= \bar{X}_K^{(DP)} \geq \theta \text{ (using equation (5.8) \& 5.9)} \\ L_3 &= \theta \frac{1}{W_{AH}} \sum_{K=t_p}^{t_p - W_{AH}} I_{\{X_K > \theta\}} \\ &= \theta \cdot f_\theta \\ &< \theta \text{ as } f_\theta < 1 \end{aligned} \tag{5.12}$$

$$\text{Also, if } X_K > \theta,\ L_3 = \theta,\ L_3 < L_1 \text{ and } L_3 < L_2$$
$$\text{as } L_1 = \bar{X}_K^{(RP)} > \theta \text{ and } L_2 = \bar{X}_K^{(DP)} > \theta \tag{5.13}$$

$$\begin{aligned} \text{Now it follows from Equation (5.4) that } \lambda_{t_p} &= \alpha_1 L_1 + \alpha_2 L_2 + \alpha_3 \theta \\ \text{Therefore using 5.12 and 5.13 above, } \lambda_{t_p} &< \max\{\bar{X}_K^{(RP)}, \bar{X}_K^{(DP)}\} \end{aligned}$$

If $\bar{X}_K$ is assumed to be same for some large window sizes, then $\lambda_{t_p} < \bar{X}_K$. It is evident from the above properties that λ is bounded above by the average values but they are under certain restricted values of X_K. Therefore, it is essential to find bounds under normal range of X_K which assists in finding the values of various windows and α easily, and is stated as a theorem below.

Theorem 5.1 *For sufficiently large W_{AH} compared to W_{RP} and W_{DP},*

$$\lambda_{t_p} \approx \alpha_1 \bar{X}_K^{(RP)} + \alpha_2 \bar{X}_K^{(DP)} + \alpha_3 \theta f_\theta^{(R)}$$

Proof: Finding the upper and the lower bounds of λ_{t_p} leads to the proof of this theorem. L_3 spans over the window of length W_{AH}, by splitting the window W_{AH} into W_{RP},W_{DP} and W_R, L_3 in Equation (5.3) can be written as

$$L_3 = \frac{\theta}{W_{AH}} \left[\sum_{K=t_p}^{t_p - W_{RP} + 1} I_{\{X_K > \theta\}} + \sum_{K=t_p - W_{RP}}^{t_p - W_{RP} - W_{DP} + 1} I_{\{X_K > \theta\}} + \sum_{K=t_p - W_{RP} - W_{DP}}^{t_p - W_{AH} + 1} I_{\{X_K > \theta\}} \right] \tag{5.14}$$

Now by equation (5.4), λ_{t_p} can be re-written as

$$\begin{aligned}\lambda_{t_p} &= \alpha_1 \frac{\theta}{W_{RP}} \left[\sum_{K=t_p}^{t_p - W_{RP}+1} X_K \right] \\ &+ \alpha_2 \frac{\theta}{W_{DP}} \left[\sum_{K=t_p - W_{RP}}^{t_p - W_{RP} - W_{DP}+1} X_K \right] \\ &+ \alpha_3 \frac{\theta}{W_{AH}} \left[\sum_{K=t_p}^{t_p - W_{RP}+1} I_{\{X_K > \theta\}} + \sum_{K=t_p - W_{RP}}^{t_p - W_{RP} - W_{DP}+1} I_{\{X_K > \theta\}} + \sum_{K=t_p - W_{RP} - W_{DP}}^{t_p - W_{AH}+1} I_{\{X_K > \theta\}} \right]\end{aligned} \tag{5.15}$$

By rearranging the terms, using W_R,

$$\begin{aligned}\lambda_{t_p} &= \underbrace{\frac{\alpha_1}{W_{RP}} \left[\sum_{K=t_p}^{t_p - W_{RP}+1} \left\{ X_K + \frac{W_{RP}}{\alpha_1} \frac{\alpha_3 \theta}{W_{AH}} I_{\{X_K > \theta\}} \right\} \right]}_{L_1'} \\ &+ \underbrace{\frac{\alpha_2}{W_{DP}} \left[\sum_{K=t_p - W_{RP}}^{t_p - W_{RP} - W_{DP}+1} \left\{ X_K + \frac{W_{DP}}{\alpha_2} \frac{\alpha_3 \theta}{W_{AH}} I_{\{X_K > \theta\}} \right\} \right]}_{L_2'} \\ &+ \underbrace{\frac{\alpha_3}{W_R} \left[\sum_{K=t_p - W_{RP} - W_{DP}}^{t_p - W_{AH}+1} \left\{ \frac{W_R \theta}{W_{AH}} I_{\{X_K > \theta\}} \right\} \right]}_{L_3'}\end{aligned} \tag{5.16}$$

For brevity, L_1', L_2' and L_3' are defined in the above equation. Now, the terms L_1' and L_2' are lower bounded by their respective averages as given below

$$L_1' \geq \alpha_1 \bar{X}_K^{(RP)} \text{ (using equations (5.16) and (5.8))} \tag{5.17}$$

$$L_2' \geq \alpha_2 \bar{X}_K^{(DP)} \text{ (using equations (5.16) and (5.9)) and,} \tag{5.18}$$

$$L_3' = \alpha_3 \frac{W_R \theta}{W_{AH}} f_\theta^{(R)} \text{ (using equations (5.16) and (5.10))} \tag{5.19}$$

where, $f_\theta^{(R)}$ refers to the fraction f_θ for the window W_R computed as

$$f_\theta^{(R)} = \frac{1}{W_R} \sum_{K=t_p - W_{RP} - WDP}^{t_p - W_{AH}+1} I_{\{X_K > \theta\}} \tag{5.20}$$

Now, using equations (5.17) to (5.19) the lower bound on λ_{t_p} can be found as

$$\lambda_{t_p} \geq \alpha_1 \bar{X}_K^{(RP)} + \alpha_2 \bar{X}_K^{(DP)} + \alpha_3 \frac{W_R \theta}{W_{AH}} f_\theta^{(R)} \tag{5.21}$$

Now, with similar algebraic manipulations the upper bound is derived as

$$L_1' \leq \alpha_1 \bar{X}_K^{(RP)} + \alpha_3 \frac{W_{RP}\theta}{W_{AH}} \text{ (using equations (5.16) and (5.8))} \tag{5.22}$$

$$L_2' \leq \alpha_2 \bar{X}_K^{(DP)} + \alpha_3 \frac{W_{DP}\theta}{W_{AH}} \text{ (using equations (5.16) and (5.9)) and} \tag{5.23}$$

$$L_3' = \alpha_3 \frac{W_R\theta}{W_{AH}} f_\theta^{(R)} \text{ (using equations (5.16) and (5.10))} \tag{5.24}$$

The indicator function is assumed to return 1 for the duration of the first two windows for the sake of simplification. Thus an upper bound for λ_{t_p} is given by

$$\lambda_{t_p} \leq \alpha_1 \bar{X}_K^{(RP)} + \alpha_2 \bar{X}_K^{(DP)} + \alpha_3 \frac{W_{RP}\theta}{W_{AH}} + \alpha_3 \frac{W_{DP}\theta}{W_{AH}} + \alpha_3 \frac{W_R\theta}{W_{AH}} f_\theta^{(R)} \tag{5.25}$$

The above relation clearly brings out the dependence of λ_{t_p} on the ratios $\frac{W_{RP}}{W_{AH}}$, $\frac{W_{DP}}{W_{AH}}$ and the value of $f_\theta^{(R)}$. To distinguish between a speaker who are active for a short duration with a speaker who is active for some time now, it is necessary to have the activity horizon window larger than recent past plus distant past window. Therefore, with $\alpha_3 < 1$,

$$\text{If, } W_{RP} \ll W_{AH}, \text{ then, } \alpha_3 \frac{W_{RP}}{W_{AH}} \approx 0$$

$$\text{Similarly, if, } W_{DP} \ll W_{AH}, \text{ then, } \alpha_3 \frac{W_{DP}}{W_{AH}} \approx 0$$

$$\text{and } \frac{W_R}{W_{AH}} \approx 1 \text{ (using equation (5.7))}$$

Thus,

$$\lambda_{t_p} \approx \alpha_1 \bar{X}_K^{(RP)} + \alpha_2 \bar{X}_K^{(DP)} + \alpha_3 \theta f_\theta^{(R)} \quad \blacksquare \tag{5.26}$$

Now, by fixing a sufficiently large value of W_{AH}, better manoeuvrability can be achieved for λ_{t_p} which assists in distinguishing speakers. The above equation also expresses the dependence of λ_{t_p} on (θf_θ). The advantage of this final expression is that the recent past and distant past windows are decoupled with activity horizon. One aspect to be clarified yet is the effect on λ_{t_p} of the recent past window vis-á-vis distant past window. Though both these windows contribute to λ_{t_p} through $\bar{X}_K$, it can be clearly seen that α_1 and α_2 fix the predisposition of λ_{t_p}. Expression 5.26 aids the selection of various windows and α's depending on the perceived requirements.

5.8.3 Computational complexity

λ is calculated using moving window averages. Therefore it is required to allocate some memory to store past X_K's. In VoIP usually packets containing 10ms - 40ms of speech

is used. As the windows correspond to a few seconds, the memory length is about a few hundred packets. By using this memory, the computation complexity λ can be made independent of the size of the window W. For the initial transient period when the windows are not completely filled, the averages L_1, L_2 and L_3 are directly calculated. Later, the memory to store X_K can form a circular buffer using a pointer and *mod* function. The pointer $p*$ on circular buffer is used to distinguish between the newest and oldest values of X_K. The pointer always moves in one direction and always points to the oldest X_K. There is a need for sorting the past values of $\sum X_K$ for each window.

Every time a new packet arrives, the difference between the new X_K and the stored oldest X_K in the circular buffer is calculated before flushing out the old value to make way for the new X_K. After storing X_K, the pointer moves to the next location that points to the oldest value. By using the difference between the oldest and the newest values of X_K, and the stored previous value of $\sum X_K$, with simple computation, $\sum X_K$ can be updated. Later, by dividing this by the window size, for example W_{RP}, $\bar{X}_K^{(RP)}$ is computed. The total number of additions and multiplications are 12 and 10 respectively independent of W for calculating the λ. Thus the calculation of loudness number is very simple for implementing in real time with the only requirements of a circular buffer management and three pointers for the windows.

The computational requirements include the pointer management and comparisons as well. The reduction in number of additions and multiplications is because the outgoing packet amplitude X_K is subtracted and the new one is added to the sum at each instant for each window. Therefore for a window one set of difference calculation, and summation to get $\sum X_K$ is required without depending on the size of the window. Hence the complexity of λ computation is $O(constant)$. Loudness number calculation is simple and comparable with generic exponential (λ_e) and L-Code methods. λ is used on a pilot implementation of conference tool with highly encouraging results with improvement in the quality of conference.

5.8.4 Fairness and resilience

These two properties have been defined in Section 3.5.4. Parameter λ has some memory depending on the spread of the windows. After a participant becomes silent, another can take the floor. As there is more than one floor, interruption is enabled. A loud participant is more likely to be heard because of an elevated λ. This ensures fairness to all participants who try to get noticed. After all, even in a face-to-face conference, a more vocal speaker gets more attention.

As speech activity reduces, λ should decrease. In case a participant misbehaves,

i.e., speaks without complying to the indentures of a conference, then a controller such as CP may take care of the situation. With participants behaving decently, loudness number is good for handling turn-taking with more than one floor. As λ is defined over finite windows, the effect of a noise burst is limited in time. Memory is overwritten beyond the window, thus returning to the ambience of current speech activity. Since averaging is done over a window, any noise burst is muffled.

Windows for λ computation are of the order of a few seconds. Thus λ varies slowly compared to communication delay, which is of the order of a few hundred ms between selectors. Therefore the distributed Algorithm 3.1 running at each selector would eventually select the same set of participants. In case the λ of participants are varying relatively fast during turn-taking, it follows from conversational analysis that repairs would intervene to control the situation (Section 4.5.3). Thus after repair the set of selected participants would be unaltered. During the brief interval of repair intelligibility of mixed speech is poor. Hence it does not matter whether the observed poor intelligibility is due to mismatch in the finally selected set S (Section 3.5.3.3) or due to multiplicity of streams.

5.8.5 Subjective quality

A qualitative study was carried out on the proof of concept setup developed in-house. The setup was tested on the same sub-network within a LAN. The motive[7] of this study was to ascertain the users' response to this implementation using λPacket loss was negligible as all the participants were located on a single LAN. At the end of the conference participants were asked four questions. (1) How good was the quality of the conference? (2) How effectively did the participants get a chance to speak? (3) Were they able to handle overlapping speech (Turn-taking)? (4) How would they compare this solution of impromptu speech and dynamically controlled speaker selection with a tightly floor controlled conference where participants took permission before speaking?

The first question encompasses the general quality of packetised speech as well as the conference application. The next three questions deal with the conversational feel of the conference (Section 4.5). Table 5.1 gives the majority evaluation. The study was undertaken with ten participants.

To the first question, marginal majority response was that the conference quality was good (grade **A**) and next majority participants said **B** due to some delay they

[7] I gather, young man, that you wish to be a Member of Parliament. The first lesson that you must learn is, when I call for statistics about the rate of infant mortality, what I want is proof that fewer babies died when I was Prime Minister than when anyone! - Winston Churchill.

Table 5.1: **Subjective quality of a conference**

Conference	Quality of Conference	Getting a chance to speak	Handling overlapped Speech	Conference using Loudness Number *vis-à-vis* Tightly Controlled Conference[a]
Conf 1	**A**	**B**	**B**	**Better**
Conf 2	**B**	**B**	**B**	**Better**

A: Good/Easy, **B**: Fair/Manageable,
C: Poor/With some problems, **D**: Bad/unacceptable

[a] Choices were: **Better, Equally good, Can't Say** and **Worse**

reported on first trial. In the second trial majority said **B**. To the second question majority of participants said that they had a fair chance (grade **B**) to speak impromptu, i.e., to access a floor. Many participants felt it was easy in both the trial runs. To the third question of handling the overlapping speech for turn-takings the answer was **B**, i.e., manageable in both the trials. However, the grades were partially spread across in first trial. To the last question, "which system is better?" almost everyone said that the dynamically controlled speaker selection system is better where a chance of interrupting the current speaker is provided.

5.9 Limitations

Human perception of hearing is a complex non-linear phenomenon [174, 163, 148]. The auditory system is modelled as performing a *spectrographic* analysis of any auditory stimulus. The cochlea in the ear converts mechanical vibrations into electrical excitations [163, 175, 176]. It may be regarded as a bank of filters whose outputs are ordered tonotopically, so that a frequency-to-place transformation is effectuated. The filters closest to the cochlear base respond maximally to the highest frequencies and those closest to its apex respond maximally to the lowest. The hearing system can also be said to perform a temporal *oscillographic* analysis of the set of neural signals that originate in the cochlea in response to an auditory stimulus. This process is important for frequencies below 500Hz and it contributes to frequency resolution up to about 1.5kHz.

The perceived loudness is not only directly proportional to the physical magnitude of the signal but also to the duration and spectral content and timbre. It also

depends on many aspects like person and subject. These parameters are not quantifiable easily. Keeping the sound level stimulus fixed, the loudness of the stimulus can increase significantly when the bandwidth exceeds the critical bandwidth. This effect was demonstrated in 1957 by Zwicker, et al., [177]. This dependence of perceived loudness on frequency is called Loudness Summation. Zwicker and Scarf experimented with a bank of critical-band filters, compression, and summation effect [178]. They found that within a critical band of frequency, loudness is independent; and that when the frequency exceeds a critical value enhanced loudness is perceived. They defined a function which relates the critical bandwidth to frequency. It was hypothesised that critical bandwidth does not change with duration. However, a recent study by Vershey [179] shows that the critical bandwidth depends also on the duration of the signals. This is referred to as temporal summation of loudness.

Loudness number defined in Section 5.6 does not take into account the spectral characteristics of speech such as pitch, prosody, and loudness summation. The reasons are: (a) very high computational requirements in realtime if these parameters have to be considered; (b) duration of speech for decision making is small (10ms to 40ms, mostly); and (c) the characterisation of frequency effect in a conference scenario is unknown. Here the effect of temporal dependence of loudness summation is negligible as the duration of averaging is in seconds. Thus temporal effect is averaged out during the computation of λ.

For instance, the tone-fall at the end of utterances in English can be analysed to add to the λ calculation. The reason is that speakers usually have a falling inflection if they are giving up the floor, i.e., if they are towards the end of their speech. Utterances with declarative intonation or with any other contour sharing the phonological specification of a low tone at the end of the tail consistently decline in sub-glottal pressure, whereas utterances with "yes-no question intonation" or any other contour sharing the phonological specification of a final high tone are reported to have shown lesser declines or even increase [180]. This is called Terminal Intonation Contour, i.e., falling intonation at the end of a sentence indicating completeness. Perhaps there exist other recognizable features. If they do, they are not considered here and some of these aspects may make λ more adaptable and more intelligent.

λ can be fine-tuned easily by varying the values of windows and α's but notably at the cost of some confusion as to the effect of these parameters. Window sizes and α's are not available except for some thumb rules and empirically tested values. θ is fixed with heuristics (Section 5.8.1), which at most provides an approximate value to be used. More trial runs, by varying θ are necessary for honing onto a better estimate

for θ. Background noise would boost λ giving unjust advantage to participants who are in noisy rooms. Adaptive background noise suppression is required then. Along with the above experiments, analysis of users' perception on the quality of the conference is also required.

Another limitation of λ is that a participant who starts answering a question posed by an existing speaker (member of S) will not be heard initially since he has a low λ though increasing. There are two cases in this scenario. First, if there are already some simultaneous speakers in the conference then there is a need of repair for conversation, and this situation leads to an onset of a Repair. Second, when there are no simultaneous speakers then, since $|S| = N_{Max}$ is more than one, the participant who has just started speaking having some small λ that is building up and higher than that of others (because others are silent) automatically gets selected. However, since λ is slightly loaded against sudden temporary raise in speech and it adapts slowly, the first few milliseconds may be dropped in a few cases.

5.10 Conclusions

A new quantifier, loudness number, λ is defined for characterising the speech status of each participant in a conference. This is believed to be the first of its kind for VoIP conference service. λ is most useful in controller-less (moderator-less) conferences. By filtering at each selector based on λ, the traffic in Internet is minimised. Computation of λ is simple and is $O(constant)$. There exists a provision to fine-tune α's and window sizes to obtain desired quality of conference. If a moderator is used in a conference, the moderator should be heard whenever (s)he speaks[8]. This can be achieved by artificially increasing the loudness number of the moderator without sending any extra information along with the audio stream.

The values of α's and window sizes given in Section 5.6.2 may serve as a starting point since they are found through many pilot runs of the conference on the PoCS. A complete characterisation of quality of a conference with respect to different terms of λ requires further study. Exhaustive results with many participants in tens or hundreds are not available presently. They demand more resources, trial runs and manpower. There is no pretension to spell out the last word on the loudness number as the results are in their infancy[9], howsoever encouraging.

[8] This need was discussed in Section 4.6.6 on page 105.

[9] Any intelligent debate or idea at a nascent stage is characterised by multiplicity of view points. It is but befitting that no one is an exception to this dictum. - Anonymous

5.11 Retrospection

Studies such as this are often criticised for performance evaluation being subjective. Comparison of perceived quality with existing conference solutions is just one aspect. Allowing multiple participants to interrupt the current speaker thereby enhancing the perceived quality of a conference is another aspect that must be compared. Redenkovic's work [27, 28, 29] appears to be the only one providing for impromptu speech; it allows every one to speak. Use of conversational analysis and quantifying participants to allocate limited floors dynamically makes this scheme novel. It is believed that this approach of specifying a bound on the number of floors and a participant status quantifier constitute concrete and a pragmatic steps towards a better audio conference.

The next chapter investigates how best these ideas can be used further to enhance the operational utility of conferencing system.

Chapter 6

On Bandwidth Saving

6.1 Introduction

It is always desirable to reduce bandwidth requirement of any application and more so for realtime systems. There are many methods of reducing bandwidth in a conferencing system. A simple technique is to use intermediate mixers[27, 38, 41] described in Chapter 2. Those techniques are not used in the proposed architecture for reasons stated already. In the proposed architecture multicasting is used if supported by the network layer. Layered multicast [65] and Simulcast [66] may be used to support specific requirements of individual clients depending on the bandwidth of the link connecting them. There are some useful coding techniques such as ITU-T G.729, G.723.1, G.726 and ETSI's GSM [69, 181, 182] that reduce the traffic by an order up to 10. However, the methods mentioned above are expensive since servers/clients are to be computationally powerful. Also, some algorithms are proprietary. They may need a DSP to carryout the computations. It is interesting to investigate whether there are other ways of reducing bandwidth and traffic. This chapter is dedicated to investigation of methods of reducing bandwidth requirements.

6.2 Motivation for Bandwidth Saving

A new methodology for VoIP conferencing was discussed so far. The conference architecture uses loudness number for turn-taking for which a simple distributed algorithm was given in Section 3.6. The efficiency of algorithm 3.1 was analysed. Bandwidth utilisation was shown to be less compared to completely distributed architecture and centralized servers (Section 3.6) by grouping clients into Selector domains. It was seen that the traffic between selectors is $N_{Max}d(d-1)$, where d is the number of domains if unicast is used between selectors. Use of multicasting – assuming that the network is

multicast enabled – results in traffic $N_{Max}d$. It is less than completely distributed or centralized server scheme discussed in Chapter 3.

Yet, one may contend that sending N_{Max} packets to and from all domains is a waste of resource, as most of these streams will not be selected. If just one client is active, selecting a subset of clients in that domain is unnecessary. Significant progress is made by fixing N_{Max} and Loudness Number for the conferencing architecture based on discussions on issues and various methods in Chapter 2. The solution finally selected was designed with the consideration of many criteria and requirements given in Sections 3.1 and 3.4.2. With a large number of domains, bandwidth reduction is an important aspect that needs to be looked at first[1] before deploying the servers and using the proposed architecture. Therefore the question is whether it is possible to reduce bandwidth requirement further.

Moreover, in an interactive conference there are many instances where a speaker is silent or on pause between talk spurts. By identifying these talk spurts one can save bandwidth by not sending the silent packets.

Bandwidth reduction can be achieved in two phases. First, techniques which affect multiple streams – targeted on the group of selected streams at selectors next, techniques which affect individual selected streams.

6.3 Techniques which Affect Multiple Streams

Techniques affecting multiple streams are invariably based on mixing, elucidated in Section 2.4.5. As the requirement is to reduce traffic on WAN, working around the proposed architecture, this section cogitates on harnessing the slow varying characteristics of loudness number to reduce bandwidth requirements. Two heuristic algorithms - pessimistic and optimistic - are presented in the sequel to this effect.

6.3.1 A pessimistic algorithm

Loudness number varies slowly with time because of averaging over a few seconds of speech activity. λ of a speech stream from a participant is calculated at every packet time period called *time-ticks*. Time-ticks are usually in the range 10ms – 40ms. At every time-tick a selector makes a decision to include the packet from a stream in the set $ToOtherSelectors$. Now, consider a scenario where the lowest λ of the finally (globally across all selectors) selected streams of set S (Section 3.5.3.3 and algorithm

[1] *Murphy's Law*: Whenever you set out to do something, something else must be done first.

3.1) exceeds the λ of the most dominant stream (based on λ) of a domain (i.e., in the local set $ToOtherSelectors$). Clearly, the chances that the next dominant streams of that domain (i.e., set $ToOtherSelectors$) get selected to the set S in the next packet period is less. Therefore these streams are not sent to other selectors. Else, if the most dominant stream from this group is in S, then comparison of the lowest λ of S is carried out with the successive streams inside a domain and these successive streams are not sent. The underlying assumptions here are:

a) Only the stream with the lowest λ of the set S in the previous time-tick might decrease to a value lower than that of the dominant λ in set $ToOtherSelectors$ in a selector's domain.

b) The likelihood of top $N_{Max} - 1$ streams of set S in previous time-tick decreasing below the second dominant stream of a domain is small. This is in agreement with the above assumption.

c) If none of the streams in S is decreasing, then only the most dominant stream of this domain has an opportunity to get selected to S in the current time-tick compared to all other streams in the domain.

Therefore, in this algorithm, at each time-tick, when all the packets that are in $ToOtherSelectors$ of a domain are less, only the most dominant packet from that domain is always sent to all other selectors to explore the possibility of it being selected to the set S. Since the chances of next and further dominant packets getting into set S is low, they are withheld. If there are packets (up to N_{Max}) with λ value greater than the lowest in set S, then they will all be selected for sending. The above assumptions are then applied to the next dominant packet inside a domain if number of streams selected is less than N_{Max}.

With this scheme there may be an error in unique selection across all selector domains for one time-tick only (in case of no delay between selectors). As λ varies slowly, the error would get automatically rectified in subsequent time-ticks. In Section 5.8.4 an account of the effect of delay between selectors on the selection of members of S was given. Here too, the effect of non-conformance of variations of λ to these assumptions may introduce some mismatch in the final selection. This is similar to the effect of delay. Since it can happen only during the contest for limited floors, repairs will ensue (Section 4.5.3).

The algorithm

The above assumptions and arguments are used to refine Algorithm 3.1 to derive the following algorithm. The algorithm is applied at every packet interval (timt-tick).

Algorithm 6.1 ***A Pessimistic Algorithm for Distributed Selection***

In the first time-tick, each selector sends top N_{Max} streams based on λ to all other selectors, form set S using Algorithm 3.1
Then at each selector, REPEAT for each time-tick
{

Step 1: *Find the lowest value of λ out of N_{Max} packets from set S in previous time-tick. Set variable LN_{least} to this value.*

Step 2: *Select N_{Max} top ranked packets based on λ from its domain, that is, the set $ToOtherSelectors$.*

Step 3: *Select packets that have $\lambda > LN_{least}$.*
IF there are N_{Max} or more than N_{Max} packets with $\lambda > LN_{least}$
then send top N_{Max} to other selectors.
ELSE IF there are $N_{Max}-1$ packets having λ more than LN_{least}
then send top $N_{Max}-1$ plus the one just lower than LN_{least}
(i. e., top N_{Max}) to other selectors.
ELSE IF there are $N_{Max}-2$ packets more than LN_{least}
then send top $N_{Max}-2$ plus the one just lower than LN_{least}
(i. e., top $N_{Max}-1$) to other selectors.
$\vdots$
ELSE IF there is NO packet with λ more than LN_{least}
then send the packet which has the highest λ
in its domain.

Step 4: *Packets sent in Step 3 to other Selectors form DB_1. Packets received packets from other selectors form DB_2.*

Step 5: *For this time-tick, find global top N_{Max} streams based on λ from $DB_1 \bigcup DB_2$ to form set S.*

Step 6: *Send these N_{Max} packets in S to clients in its domain.*

}

Discussion

In this algorithm, at least one packet in each packet period from each domain is sent. This is done even if it has λ less than that of the least one of S in previous time-tick assuming that it might get selected. One expects this algorithm to be called an optimistic algorithm! However, the fear that a stream may miss out on selection, if λ has increased, is driving this algorithm. Thus saving in bandwidth may not be high. The net network traffic in a time-tick in the best case (well behaved period of a conference) is approximately $d(d-1)$ using unicast instead of $N_{Max}d(d-1)$. In domains with multicast enabled network the traffic would be $(N_{Max}+d)$ in place of $N_{Max}d$. Considerable valuable bandwidth can be saved using this heuristic algorithm. The resulting traffic complexity reduces by a factor of N_{Max} in both the cases during well-behaved periods. Since the conference is well behaved this case prevails most often.

In this algorithm the saving in traffic is at the cost of relaxing the condition of forming a globally unique set S. However, the discrepancies in selected streams at different domains remain for a short time depending on the transportation delay between any domains. Even for a total delay of 400ms, uniqueness might be lost only for a few time-ticks. In a realtime interactive conversation this duration is not perceivable by the listener. For example, if every one laughs over a joke, there will be a sudden rise in the number of packets and traffic would be upper bounded by $O(d^2)$ for a short period for unicast networks. In multicast enabled network it is $O(d)$. After the burst, the algorithm adapts itself smoothly to cut down some traffic. However, it depends on the dynamics of λ. These periods are short since repairs will intervene as argued earlier. The next subsection investigates the possibility of further reduction of traffic outside selector domains.

6.3.2 An optimistic algorithm

The following algorithm withholds all packets which have λ lower than the least λ of S. The stand point here is that transmitting even a single packet from a domain when its λ is lower than that of the least in S is unnecessary. The *optimism* here is that if a low value of λ is seen then most possibly it will remain so and is not necessary to select it immediately. If λ of packets in S is reducing and λ of some other participant increasing, the stream for which λ crosses the lowest λ of S is sent. One can find the correct and unique N_{Max} streams after some time-ticks depending on the transportation delay between domains. As the packet period is of the order of 10ms – 40ms, the error in the selection is unnoticeable.

The algorithm

The above reasoning is incorporated into Algorithm 6.1 to arrive at the following algorithm. The algorithm works at each packet intervals.

Algorithm 6.2 ***An Optimistic Algorithm for Distributed Selection***

In the first time-tick, each selector sends top N_{Max} streams based on λ to all other selectors, and form set S using Algorithm 3.1
Then at each selector, REPEAT for each time-tick
{

Step 1: *From S, find the least λ in previous time-tick. Set variable LN_{least} to this λ.*

Step 2: *Select top N_{Max} packets from its domain; that is, the set $ToOtherSelectors$.*

Step 3: *Select packets having $\lambda > LN_{least}$.*
IF there are N_{Max} or more than N_{Max} packets with $\lambda > LN_{least}$
then send top N_{Max} to other selectors.
ELSE IF there are $N_{Max}-1$ packets having λ more than LN_{least}
then send top $N_{Max}-1$ plus `the` *packet*
in case the exception is satisfied in Step 4.
ELE IF there are $N_{Max}-2$ packets more than LN_{least}
then send top $N_{Max}-2$ plus `the` *packet*
in case the exception is satisfied in Step 4.
$\vdots$
ELSE IF there is no packet with λ more than LN_{least}
then no packet is sent with the following exception.

Step 4: Exception:
IF a packet in this selector domain had LN_{least} of S in the previous interval then select that packet even if its λ has reduced to less than LN_{least} (its own value in previous time-tick) in the current time-tick.
ELSE IF the λ of packet of LN_{least} has increased in the current time-tick to a higher value then it would have already been selected in Step 3 and sent. Now, a different packet less than or equal to LN_{least} is sent to other Selectors.
In both the above cases, the number of streams selected is limited to N_{Max}.

Note: The exception occurs only at that selector which had sent the packet that had least λ in S for the previous time-tick.

Step 5: *Packets sent (in step 3 & 4) to other Selectors form* DB_1. *Packet received from other Selectors form* DB_2.

Step 6: *Now for this time-tick, find global top* N_{Max} *packets, based* λ *from* $DB_1 \bigcup DB_2$ *to form* S.

Step 7: *Send these* N_{Max} *packets in* S *to clients in its domain.*

}

Discussion

This algorithm is very similar to the pessimistic algorithm. The difference is in not sending packets with λ less than the least λ of S in the previous time-tick. The least λ of S is to be tracked always since it is the reference. This is taken care of in Step 4. For one time-tick the packet which had the least λ in S at the previous time-tick would survive even if its λ is less than that of a packet in some other selector domain. This is not for more than one time-tick. There would be some inconsistency. However this inconsistency is during floor change; it is not perceptible to a user.

This algorithm restricts the number of packets on network to N_{Max}. Most of the time, there will be no more than N_{Max} streams in the network during conduct of a well-behaved conference. In the best case (well behaved period of a conference) total traffic is of constant order (i.e., totally N_{Max} packets) for multicast enabled network. Net traffic is $N_{Max}d$ for unicast networks in the best case. Exceptions (say, when everyone laughs), introduce sudden burst of traffic at most of $O(d^2)$, for unicast network, as explained earlier. The algorithm adapts itself later to cut down some traffic. It is however dependent on the dynamics of λ. As before, due to repairs these situations should be for a short duration. Saving will be better than the pessimistic algorithm even during this scenario. The non-uniqueness of streams selected across selectors during turn-taking introduces some inconsistencies. Again, users do not perceive its effect as explained earlier in connection with Algorithm 6.1. These advantages are due to exploitation of the characteristics of λ.

The above algorithms save bandwidth and computation at each selector and lead to a scalable architecture with multiple selectors mainly because clients are grouped in domains. The operating bandwidth is not always dependent on the total number of active clients with these algorithms because of intelligent filtering. Yet, there are a few occasions where a sudden surge in total traffic is witnessed.

The prospect of reducing traffic further is considered in the next section. With these algorithms bandwidth saving in large conferences is expected to be high.

6.4 Techniques which Affect a Single Stream

Bandwidth utilisation can be optimised at the level of individual streams. In literature there are many methods that are applied to a single stream. Two of them are briefly discussed here. They are based on *filtering* out high bandwidth data streams before the traffic enters narrow bandwidth links. The use of self-propagating filters over a dissemination tree is proposed by Pasquale, et al. [183]. Leaf nodes specify their requirements to the filter nodes above them. Filter nodes convert an incoming stream with a low bit rate coding scheme referred to as trans-coding. When a non-leaf node has multiple output links with similar filters, the filter is propagated to a node higher up in the tree. Multiparty Multimedia Control (MMC) has been proposed in [184], in which at merge points in the tree, filters of the same types are combined and further propagated. This scheme requires full knowledge of the distribution tree topology and processing capabilities at each node to achieve optimal network utilisation with minimal processing. Evidently, in a conferencing solution, due to multiple simultaneous speakers and frequently changing set of speakers this scheme is difficult to implement. These schemes require a moving entity such as "filters" other than selectors. This considerably burdens the network entities such as routers for configuring the filters.

Proxies [185] can also be used for filtering multiple streams. Proxies are media gateways situated at strategic points in the network. They transform media streams according to the clients' requirements. By placing a proxy between the source and the sink of data, network bandwidth variation can be accommodated through format 'distillation', and allocation of bandwidth across flows can be optimised using intelligent rate adaptation. Moreover, the proxy can translate streams between different media, enabling communication between otherwise incompatible clients. Proxies are at the edges and do not work with the core networks.

None of these techniques address the issue of simultaneous speakers. Therefore, the amount of audio data and processing for these techniques increases proportionally with the number of simultaneous speakers. The selector proposed in Chapter 3 might be thought of as a functional proxy. Filters and proxies discussed above work closely with the network. Selectors works at the application layer emulating ALF (Section 3.8.4). Thus there is a need for a technique that works independently, in the application layer, without a requirement to interact with lower layers. A simple method that is

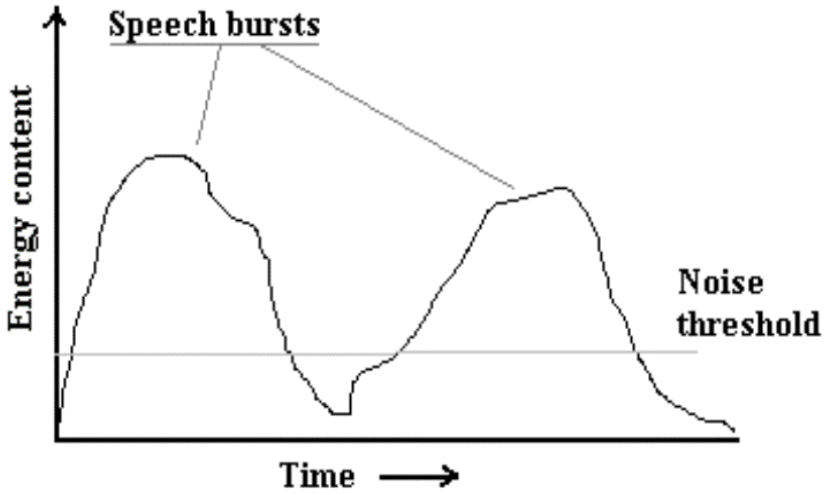


Figure 6.1: **Speech bursts**

independent of the network and can be easily implemented at a participant's terminal is presented below.

6.5 Voice Activity Detection (VAD)

This section investigates bandwidth saving on individual streams after a speech segment is recorded at the application layer. Here the problem of reducing bandwidth for a voice connection on Internet using Voice Activity Detection (VAD) is addressed, while maintaining the voice quality. Conversational speech is a sequence of contiguous segments of silence and speech (Figure 6.1) [186]. The strategy is to use the fact that no speech channel (individual stream) is continuously active. When there are conversations each party is active, typically, for less than 50% of the time [187]. Kaleed, et al., report a 40% activity of speech in VoIP [188]. In fact even during active periods there are times when sizeable pauses between words and expressions exist [189]. Thus VAD algorithms take recourse to speech pattern classification to differentiate between speech and silence periods to save bandwidth. Identifying and rejecting transmission of silence periods helps reduce traffic.

A speech segment may be classified as an ACTIVE or an INACTIVE (silent or non-speech) segment based on its energy. Characteristics of speech other than energy may be profitably taken into account before deciding this (Sections 5.2.1 and 5.2.2). The term 'INACTIVE segment' or 'silent segment' refers to a period of incomprehensible sound which may not have zero-energy. Therefore VAD algorithms are required to

tackle periods having low audible speech.

VAD is also used in Speech Recognition systems, Compression and Speech coding [190, 191, 192, 193, 194] to find the beginning and end of talk spurts. These are non-realtime applications. G.729 Annex B [69] and GSM [182] coding techniques use inbuilt VAD; they are computationally expensive. Pavel Sovka and Peter Pollak have reported some work using spectral subtraction [195] and cepstrum [196, 197] mainly for speech enhancement systems. They use *overlapping frames* for detection. VAD involving complex higher order statistics (HOS) is proposed in [198]. Algorithms based on Mel-Frequency Cepstrum Coefficients (MFCC) with Bayesian adaptive methods is in [199] and based on Hidden Markov Model (HMM) in [200]. These are computationally complex and require training and building a model. These algorithms are mainly used in speech recognition and speech enhancements.

VAD is very useful in VoIP, where speech activity detection is needed to reduce traffic on the network. It is less demanding than the above applications and is more in favour of decision making in realtime. Actually, there is no great increase in bandwidth if a few silent packets are marked as speech segments.

The packet size (payload size) in VoIP systems is determined taking into consideration, delay, throughput and Maximum Transfer Unit (MTU) on the path of the links. If a packet does not contain voice signal it need not be transmitted. VAD for VoIP has to determine if a packet contains useful voice signal. The decision by VAD algorithms for VoIP is always on a packet-by-packet basis.

6.5.1 Requirements of a VAD algorithm

The following are some desirable aspects of VAD algorithms for VoIP applications.

(1) A good decision rule that exploits the properties of speech to consistently classify segments of speech into INACTIVE and ACTIVE segments.

(2) Adapting to non-stationary background noise enhances robustness.

(3) The computational complexity of VAD algorithm must be low to suit real-time applications.

(4) Toll quality voice should be achieved after applying VAD algorithm.

(5) Maximising the detection of INACTIVE periods to save bandwidth.

A VAD algorithm should achieve all of the above requirements. The assumptions on which the VAD algorithms proposed here are based on are as follows [189]:

A. Speech is quasi-stationary. Its spectral form changes over short periods, e.g., 20ms – 30ms.

B. Background noise is relatively stationary, changing very slowly with time.

C. Energy of the speech signal is usually higher than background noise energy; else speech will be unintelligible.

In the following sections, various easy to implement VAD algorithms with varied complexity and quality of speech are presented. Mainly, time and frequency domain techniques are discussed. Results and comparison of various algorithms with qualitative assessment of speech quality are presented.

6.5.2 Parameters for VAD design

A VoIP packet may be fragmented into possibly smaller contiguous chunks of speech segments referred to as *frames* hereafter. A frame containing speech is labelled as ACTIVE. A silent frame is labelled as INACTIVE. A parameter is associated with each frame. These parameters can be energy of a frame, Zero Crossing Rate (ZCR) and variance of energy across a few past frames. If this parameter exceeds a certain threshold, the frame is classified as ACTIVE; else it is INACTIVE. Before making the final decision that speech is not present, a number of consecutive inactive frames have to be detected. This is determined by the length of the hangover time which is of the order of 60ms – 100ms or $6 - 10$ frames in this case [189]. If, after hangover time, the stream still remains inactive, then the frame is classified as 'INACTIVE' and is communicated to the higher layers from VAD algorithm. This hangover is not explicitly shown in the algorithms discussed below.

Choice of frame duration

VoIP receivers queue up the incoming packets in a packet buffer that allow them to play audio even if incoming packets are delayed due to network conditions. Consider a VoIP system having a buffer of 7 to 10 packets. Having packet duration of 10ms allows the VoIP system to start playing the audio at the receiver's end after 30ms to 40ms from the time the queue started building up but for higher size packets it would exceed this duration. However, many other factors influence packet size selection. Current VoIP systems use 10ms to 40ms packet size. Therefore the VAD algorithms here use a generic value of 10ms frame duration. If a larger size packet is to be classified then

the VAD algorithm is applied repeatedly on the frames contained in that packet. A packet is always classified as ACTIVE if at least one frame in it is active.

At $8kHz$ sampling, a 8ms (64 Samples = 2^6) frame size may be used for frequency domain algorithms, to avoid padding in Discrete Cosine Transform (DCT) calculations used in VAD algorithms. Nonetheless, using 8ms frames introduces padding for the last frame if VoIP application is designed with 10/20/30ms periods per packet. Otherwise, the VoIP packet will have to miss some input signals. Therefore, padding becomes inevitable in these circumstances. For 40ms packet size used in the implementation (as a part of this thesis), five 8ms frames make a packet without necessitating padding.

Coding

All VAD algorithms described here uses input speech signal recorded with a sampling frequency of $8kHz$ and 256 levels of linear quantisation (8 bits PCM) and single channel recording. A different quantisation level may be used if required, with some variations in the algorithms. If some other coding is used it should be converted to PCM, since the basic properties of speech signals are defined here on PCM samples only. With 8 bit linear PCM coding, a frame duration 10ms corresponds to 80 samples for time domain algorithms.

Energy of a frame

Energy of a frame indicates possible presence of speech. Let $X(i)$ be the i^{th} sample of a speech stream. If the frame has n samples, then the j^{th} frame can be represented in time domain by a sequence as

$$f_j = \Big\{ X(i) \Big\}_{i=(j-1)n+1}^{jn} \tag{6.1}$$

Energy E_j associated with the f_j is

$$E_j = \frac{1}{K} \sum_{i=(j-1)n+1}^{jn} X^2(i) \tag{6.2}$$

and the frequency coefficients for f_j are found using DCT as

$$F_j(f) = DCT\Big\{ f_j \Big\} \tag{6.3}$$

DCT of $x(i)$ is defined as

$$X[k] = \sum_{i=1}^{N} x(i) \frac{cos(2i+1)\pi k}{2N}, \quad k = 1, 2, \ldots, N$$

Initial value of threshold

A frame has to be classified as an ACTIVE or INACTIVE taking into consideration the energy of background noise. For this, energy of each frame is compared with a *threshold*. A fixed threshold would be 'deaf' to varying acoustic environments of the speaker. The scheme must have the wherewithal to adapt the threshold online and in realtime. Initial value for the threshold is important. An arbitrary initial choice of the threshold may cause deteriorated performance since it may take a long a time to approximate the background noise iteratively. Two methods are proposed for finding a starting value for the threshold.

Method 1: The VAD algorithm is trained for a small period (a few frames) by a pre-recorded sample that contains only background noise. The initial threshold for time and frequency domain techniques are computed from these samples. The initial estimate of threshold is obtained by taking the mean of the energies of each sample as,

$$E_r = \frac{1}{m}\sum_{i=0}^{m} E_i \tag{6.4}$$

where, E_r is initial reference estimate and m is the number of frames in the pre-recorded sample. In all algorithms some form of energy threshold is used except in one where the variance of frequency spectrum is used. The initial reference estimate for variance of spectrum is

$$\sigma_r = \frac{1}{m}\sum_{i=0}^{m} VAR\Big\{F_i(f)\Big\} \tag{6.5}$$

where, variance computation is across the frequency spectrum of a frame and is obtained using Equation 6.3. In this method a pre-recorded sample of 5 secs, (500 frames in time domain and 625 frames in frequency domain) is used.

Method 2: Though similar to the previous method, here it is assumed that the initial 100ms of the sample does not contain any speech; i.e., these initial 10 frames are considered INACTIVE [201]. It is observed that after the call establishment users require at least about 200ms to recognise the call establishment and respond. The mean reference estimate is calculated using Equations 6.4 and 6.5 with $m = 10$. This method is very useful in VoIP environments as it can compute background noise on-the-fly whenever a call is started. However, this method may not be as accurate as the first method since fewer frames are used for computation. All algorithms proposed here use the second method for initial approximation for E_r or σ_r. Threshold E_t for comparison is computed as $E_t = kE_r$ where $k > 1$. Six algorithms, including simple energy based algorithm [201], for VAD are given below.

6.6 VAD Algorithms - Time Domain

Energy of a frame is the basis for all time domain VAD algorithms since the energy of ACTIVE frames is higher than that of INACTIVE frames [186]. The decision that j^{th} frame is ACTIVE or INACTIVE is given below.

$$\begin{gathered} IF(E_j > kE_r) \text{ Frame is ACTIVE} \\ \text{Else, Frame is INACTIVE} \\ \text{where } k > 1 \end{gathered} \tag{6.6}$$

Having a scaling factor k, allows a safe band for the adaptation of E_r and in turn the threshold E_t. Without this safe band, E_r will not be updated when background noise increases. The following algorithms use Decision 6.6.

6.6.1 Linear Energy-Based Detector (LED)

This algorithm is similar to the one in [202, 201] where energy of a frame is compared with a threshold. A frame is decided as ACTIVE or INACTIVE as in Decision 6.6, with $k = 2$. Care must be taken to update the reference energy E_r as and when the energy of background noise varies. That is E_r must be adaptive. It is now sufficient to specify E_r (Decision 6.6) to formulate the scheme completely. The same update rule is used in other algorithms that follow next.

Computation of E_r: Since background disturbance is relatively non-stationary an adaptive threshold is more appropriate. This rule to update the threshold value is found in [202, 197].

$$E_{rnew} = (1-p)E_{rold} + pE_{noise} \tag{6.7}$$

Here, E_{rnew} is the updated value of reference for threshold,

E_{rold} is the previous reference value for threshold,

E_{noise} is the energy of the most recently detected inactive frame.

p $(0 < p < 1)$ is chosen considering the impulse response of the System 6.7. Taking Z-Transform,

$$E_r(z) = (1-p)z^{-1}E_r(z) + pE_{noise}(z) \tag{6.8}$$

The transfer function is

$$H(z) = \frac{E_r(z)}{E_{noise}(z)} = \frac{p}{1-(1-p)z^{-1}} \tag{6.9}$$

The impulse response for $H(z)$ shows a decaying curve. Where p is set to 0.2 experimentally. It is observed that $p = 0.2$ is approximately equivalent to an effective

rectangular window of length $1/p$ frames, i.e. 50ms. In effect, the calculation for E_{rnew} is influenced by most recent, past 5 INACTIVE frames. Usually, pauses between two syllabi is greater than 80ms and phoneme length is on an average 80ms [77]. Thus effectively even between syllabi the algorithm is able to update E_r. However, it is subjected to hangover time.

> **Remark**: Calculation of new E_r is the basis for all the other algorithms that follow with certain modification to get improvement in perceptual speech quality. $p = 0.2$ is used for all algorithms hereafter.

Merits and demerits of LED

This algorithm is very simple to implement. It gave an acceptable perceptual speech quality after compression. Clipping occurs with a large k, whereas small k results in less compression. Experiments reported here used $k = 2$ where as Pollak [195, 197] uses a value of $k = 1.5$. This algorithm did not give a good speech quality under varying background noise. While Equation 6.7 renders adaptability to the algorithm, it makes the threshold sluggish. Consequently the algorithm is incapable of effectively tracking rapidly changing background noise. This leads to undesirable speech clipping, especially at the beginning and end of speech bursts. Bursts in plosives were clipped. Fricative sounds (see Section 5.2.1 on characteristics of speech) as in the words such as "high" and "flower" were clipped completely. This is because the algorithm is based exclusively on the energy content of the frames. Low SNR conditions cause undue clippings, thereby deteriorating performance. The results of the algorithm on test speech templates are given at the end of these discussions in Section 6.8 for a comparative study.

6.6.2 Adaptive Linear Energy-Based Detector (ALED)

The sluggishness of LED is a consequence of p in Equation 6.7 being insensitive to the noise statistics. For this E_r based on second order statistics of INACTIVE frames is computed. A buffer of the most recent m INACTIVE frames is maintained. In these experiments m is set to 20 empirically. Buffer A contains the value of E_{noise} rather than the voice packet itself. $A(1)$ has energy of the latest inactive frame, A(2) of the one immediately previous to A(1), and so on. Whenever a new inactive frame is detected it is added to the queue and the oldest one is removed (Figure 6.2). The variance of the buffer in terms of energy is given as,

$$\nu = VAR\Big\{A(i)\Big\}; \quad i = 1, 2, \ldots, m \tag{6.10}$$

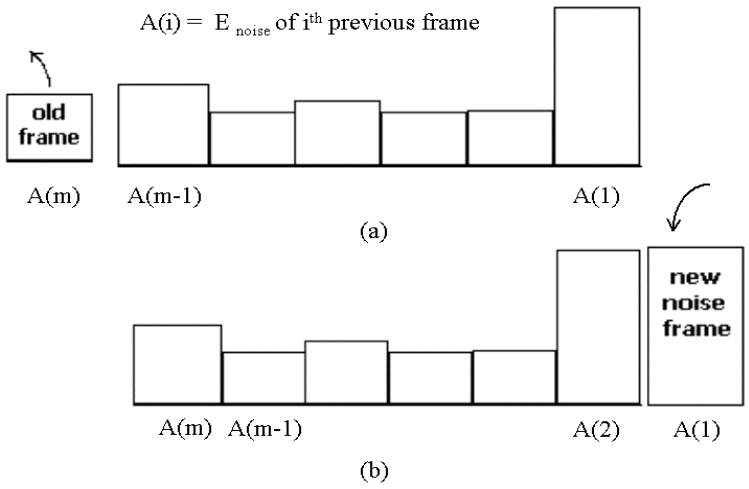


Figure 6.2: **Buffer contents before and after addition of new frame**

In Equation 6.10, unlike in Equation 6.7, E_{noise} represents the energies of the frames in the noise buffer. A change in the background noise is reckoned by comparing the energy of the new INACTIVE frame with a statistical measure of the energies of the past m INACTIVE frames. Consider the instant of addition of a new INACTIVE frame to the noise-buffer. The variance just before addition (Figure 6.2(a)) is denoted by ν_{old}. After addition of the new INACTIVE frame (Figure 6.2(b)), the variance is ν_{new}. A sudden change in the background noise as shown in Figure 6.2 would mean,

$$\nu_{new} > \nu_{old} \tag{6.11}$$

A new rule to set p in Equation 6.7 is given in Table 6.1 (see LED algorithm to choose the range of p). Here, p is not constant as in Equation 6.7. Better adaptability stems from the fact that E_r is reconciled using p, in addition p is also adjusted according to the background noise. With this, sluggish E_r is made to adapt faster to changes in background noise. Detection is still energy-based. Heuristic values of p, for various ratios given in Table 6.1 were found experimentally.

Merits and demerits of ALED

ALED is an improvement over LED as it provides higher degree of adaptability while retaining the realtime performance of energy-based algorithms. Adaptability to the background noise is better compared to LED. The choice of p determines the pliancy of the threshold. The algorithm is more capable of eliminating noise conditions like

Table 6.1: **Dependence of p on $\frac{\nu_{new}}{\nu_{old}}$**

Range	value of p
$\frac{\nu_{new}}{\nu_{old}} \geq 1.25$	0.25
$1.25 \geq \frac{\nu_{new}}{\nu_{old}} \geq 1.10$	0.20
$1.10 \geq \frac{\nu_{new}}{\nu_{old}} \geq 1.00$	0.15
$1.00 \geq \frac{\nu_{new}}{\nu_{old}}$	0.10

those that develop on switching on a fan in mid conversation. However, ALED like LED is incapable of reliable detection of bursts in plosives and fricatives. Low SNR conditions caused undue clippings of the speech, as in LED algorithm. It is a poor performer under low SNR condition.

6.6.3 Weak Fricatives Detector (WFD)

LED and ALED are exclusively energy-based. High energy voiced speech frames are always detected in all VAD algorithms even under low SNR conditions. Low energy unvoiced speech is commonly missed [202], thus reducing perceptual speech quality. Some of the missed ones are [202] (see Section 5.2.1 for different types of speech sounds),

(i) weak fricatives (/f, th. h/) at the beginning or end of an utterance,

(ii) weak plosive bursts(/p, t, k/),

(iii) final nasals,

(iv) voiced fricatives at the end of words which become devoiced and

(v) trailing off of certain voiced sounds - e.g., "three" and "binary", final /i/ becomes unvoiced.

The algorithm in this section is designed to overcome this problem. ZCR for a signal is defined as the number of times that the signal crosses the line of 'mean value' or 'zero line' per frame. ZCR [163, 201] for a speech signal lies in a fixed range. ZCR for vowels is very low, less than 8 per $10msec$ frame in most cases. ZCR for noise is unpredictable.

For fricatives, ZCR is approximately in between 15 and 30. Douglas [163] puts this range between 14 and 49. Since the recording is done with $8kHz$ sampling rate (band limiting signal to $4kHz$), observed zero crossings were less. In most of the algorithms, ZCR above min$\{25, ZCR$ of noise frames$\}$ [201] are taken as fricatives though they pass high frequency noise frames as speech frames. The technique here is to set an upper bound on ZCR compromising on a few occasions with fricatives that have higher ZCR. For background noise in a home/laboratory environment due to fans or air conditioners, ZCR is found to be around 40. To avoid this specific background noise the upper limit was set to 30. Now frames with some fricative sounds were not detected since, the limit is at the lower side of the fricatives range. The transitions from fricatives to vowels and vice versa are detected easily. The value taken here is a compromise to achieve a little more compression without allowing background noise of these specific environments. N_{ZCD} is the number of zero crossings of a signal $s(.)$ detected in a frame.

$$N_{ZCD} = 0.5|sign(s(i)) - sign(s(i+1))| \tag{6.12}$$

$$sign(s(i)) = \begin{cases} 1 & \text{if } s(i) \geq 0 \\ -1 & \text{Otherwise} \end{cases} \tag{6.13}$$

Now, the decision rule is:

$$\begin{gathered} IF(N_{ZCD} \in \Re) \text{ Frame is ACTIVE} \\ \text{Else, Frame is INACTIVE} \end{gathered} \tag{6.14}$$

Here $\Re = \{15, 16, \ldots, 30\}$ denoting ZCR per 10ms frames. It is set after experimentally observing frames of various speech samples. The decision rule 6.14 above is used in tandem with the energy based classifiers. Zero Crossing Detector (ZCD) is used only on frames identified as INACTIVE by LED. The decision is first based on energy of a frame and later on ZCR before finally deciding whether a frame is indeed INACTIVE. The decision rule is shown in Figure 6.3. It is able to detect low energy phonemes in quite a number of cases. The range empirically set here gave good response for VoIP systems having comparatively less background noise.

Merits and demerits of WFD

This algorithm is better than previous two as it may detect some frames as speech that would have been otherwise classified as INACTIVE. ZCD improves low SNR performance of the VAD algorithm. Moreover, WFD is more capable of detecting bursts of plosives and fricatives than LED and ALED. As noisy or non-speech frames may have the same number of zero crossings as in speech frames, ZCD often makes

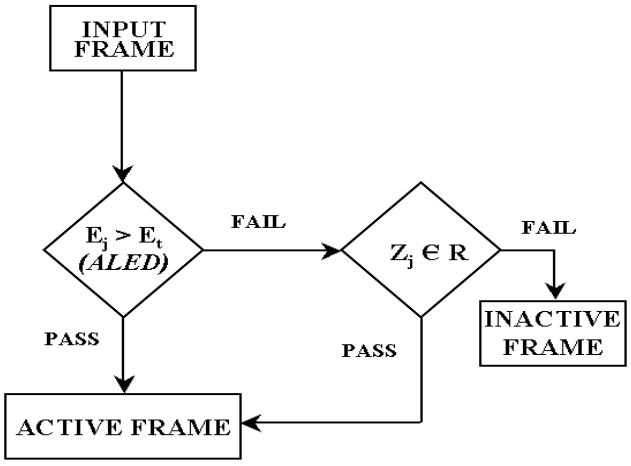


Figure 6.3: **Flowchart for WFD**

incorrect decisions. Also some of the pure fricative sounds might be missed due to the limited range of $\Re$. However, the transition of a fricative to a vowel can be detected.

6.7 VAD Algorithms - Frequency Domain

This section deals with VAD algorithms that harness frequency domain characteristics of speech signals. Frequency spectrum gives a clear picture of each frame compared to energy of a frame that was used earlier. DCT is used for spectral analysis as,
(a) It is a real transform.
(b) It has approximately twice the frequency resolution of DFT, and
(c) It is computationally less complex compared to DFT.

6.7.1 Linear Sub-band Energy Detector (LSED)

This algorithm takes its decisions based on energy comparisons of the input frames with a reference energy threshold in the frequency domain. The spectrum obtained by Equation 6.3 obtained is divided into four bands of width $1kHz$, the bands are $0-1kHz$, $1-2kHz$, $2-3kHz$, $3-4kHz$. In some cases the lower energy band is sub divided further. However, for ease of implementation the lower frequencies are considered to be in a single band and with higher priority over other bands. The energy for the j^{th} frame for band $n \in \{1,2,3,4\}$ corresponding to the above ranges

respectively is

$$E_j^{(n)}[f] = \frac{1}{B^{(n)}} \sum_{i \in n^{th} band} (F_j(f_i))^2 \tag{6.15}$$

where, $B^{(n)}$ is the number of frequency components in n^{th} band. $E_r^{(n)}$, the counterpart of E_r of energy based algorithm of a frame is calculated as above for all bands. The condition for presence of speech in each band is given by,

$$E_j^{(n)}[f] > kE_r^{(n)}[f] \text{ for } n^{th} \text{ band} \tag{6.16}$$

The thresholds are computed recursively, but for each band separately as a convex combination as in Equation 6.7. For the n^{th}band,

$$E_{rnew}^{(n)}[f] = (1-p)E_{rold}^{(n)}[f] + pE_{noise}^{(n)}[f] \tag{6.17}$$

Thus in each band the energy threshold is computed as a convex combination of the previous energy threshold and the latest noise update of the current band. The procedure to choose the value of p is same as in LED. Most of the energy in voice signal tends to be in the lowest frequency band, i.e., $0 - 1kHz$. Selective threshold comparison in the lowest band alone provides good decisions. Embedding this condition in the algorithm improves the performance of VAD. The decision rule for this algorithm is given in Figure 6.4. A frame is declared ACTIVE if the lowest frequency band is ACTIVE and any two out of the remaining three bands are ACTIVE. AND and OR gates are used in the diagram for ease of presentation.

Merits and demerits of LSED

Unlike its time domain counterparts, the LSED algorithm computes the energy content of speech across its frequency spectra. Typically lower frequencies have higher energy content. Four different thresholds adapt independently to the noise in each corresponding sub band. This adds to the adaptability of the algorithm. Another advantage of LSED is its capability to identify coloured noise unlike the time domain algorithms. Like exclusive energy based VADs, LSED also degrades in low SNR conditions to some extent and is incapable of detecting fricatives. Computational requirements are high compared to time domain algorithms.

6.7.2 Spectral Flatness Detector (SFD)

The algorithms proposed so far are inefficient in low SNR conditions. This algorithm is intended to work even at low SNR. White noise has a flat spectrum while voiced signals

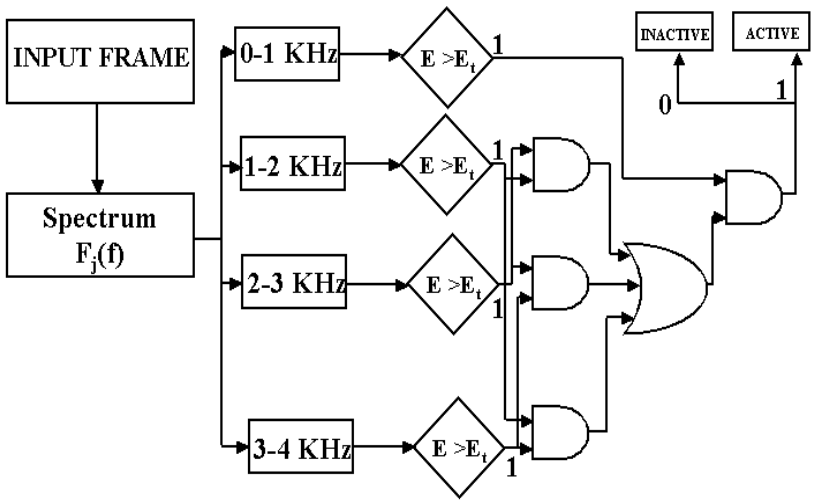


Figure 6.4: **Flowchart for LSED**

have a non-flat spectrum with more spectral content in the lower frequencies. Thus high variance implies speech content while low variance implies noise alone. Variance σ_j for the j^{th} frame is,

$$\sigma_j = VAR\Big\{F_j(f)\Big\} \tag{6.18}$$

Variance of each frame is compared against the variance threshold $k\sigma_r$ (σ_r is found using Equation 6.5) to determine its activity. The condition for presence of speech in the given frame is,

$$\begin{aligned} IF(\sigma_j > k\sigma_r) \;\; &\text{Frame is ACTIVE} \\ &\text{Else, Frame is INACTIVE} \end{aligned} \tag{6.19}$$

σ_r is updated during INACTIVE frames using the convex combination (with new, old and noise subscripts as in Equation (6.7)),

$$\sigma_{rnew} = (1-p)\sigma_{rold} + p\sigma_{noise} \tag{6.20}$$

The algorithm is similar to LED but uses σ in place of energy. The value of p is set to 0.2.

Merits and demerits of SFD

SFD uses the information contained in the spectral shape to judge the ACTIVITY of frames. Vowels and fricatives are detected well by SFD owing to their high variance.

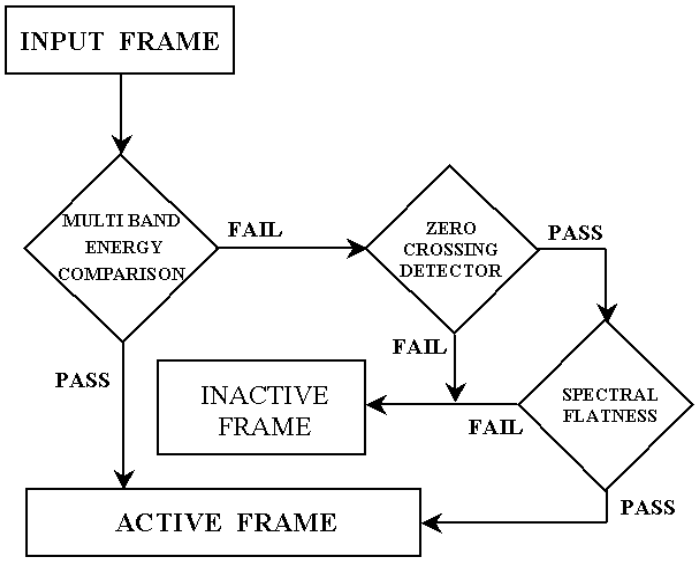


Figure 6.5: **Flowchart for CVAD**

SFD is also good at detecting the low spectral variance. However, as coloured noise also has high variance it is often wrongly judged as ACTIVE. SFD performs better than LED, ALED and LSED in low SNR conditions, as it is not exclusively energy dependent. Computational requirements are high compared to time domain algorithms.

6.7.3 Comprehensive VAD (CVAD)

In the previous algorithms only a few characteristics of speech were exploited. To obtain better perceptual speech quality, the ideas discussed earlier are all incorporated into one algorithm. This VAD algorithm is capable of identifying low spectral variance as well as frequency selective noise and maintaining a good perceptual quality of speech. The calculations of parameters for the previous algorithms remain the same but the decision rule is changed based on high priority for energy comparison. The decision flowchart for this algorithm is shown in Figure 6.5. The decision rules for individual blocks in the figure are same as in previous algorithms. Skipping ZCD and SFD once the multi-band energy comparison passes the test can reduce computation. The main aim of this algorithm is to reduce the misdetection that a ZCD might introduce by passing it through the SFD and better detection using frequency domain techniques. Many other variants can be developed with these algorithms.

Merits and demerits of CVAD

CVAD incorporates the best features of zero crossings, spectral flatness and energy-based detectors to build a robust VAD. Energy based detectors as a first level of detection ensures good compression. Zero crossings and spectral flatness help in detection of low energy fricatives. While the CVAD sometimes gives a compression figure and speech quality that is comparable to its constituent algorithms, it is observed to be more consistent and reliable. The perceptual quality of speech is better compared to other algorithms in most of the cases. The computational requirement is high compared to all the previous algorithms. SFD and CVAD are comparable in all aspects. CVAD scores over SFD with reduced percentage of misdetection.

6.8 Results and Discussions

All the algorithms discussed above were tested with test templates. The test templates used varied in loudness, speech continuity, and background noise. Both male and female voices were used. Samples were recorded in the usual VoIP environments such as offices/laboratories and homes using microphones that come with standard PCs having Creative sound blaster cards. Signals were recorded with $8kHz$ sampling and 8 bits linear PCM quantisation. To find the computation complexity, MATLAB implementations were used.

6.8.1 Indices of performance

The performance of the algorithms was studied on the basis of the following indices:

1. *Percentage compression*: It is the ratio of total INACTIVE frames detected to the total number of frames formed expressed as a percentage. A better VAD should have higher percentage compression.

2. *Subjective Speech Quality*: The quality of the samples was rated on a scale of 1 (poorest) to 5 (best) where 4 represents toll grade quality. Input signal was taken to have speech quality 5. Speech samples after compression were played to independent jurors in a random sequence for unbiased decision.

3. *Objective Assessment of Misdetection*: The number of frames which have speech content, but were classified as INACTIVE and number of frames without speech content but classified as ACTIVE are counted. The ratio

of this count to the total number of frames in the sample is expressed as a percentage. This gives a quantitative measure of VAD performance.

4. *Floating Point Operations (FLOPS) required*: The total number of FLOPS is calculated for all algorithms to compare their relative complexity. This index helps in assessing complexity of each algorithm to be implemented in realtime.

Even though percentage of misdetection could represent the perceptual quality of speech after applying a VAD technique, the quality of speech has to be assessed only by the Mean Opinion Score (MOS). Percentage of misdetection gives an approximate assessment of the performance of an algorithm. The scores presented here are taken on a realistic environment with listeners using their desktop PCs. If this work has to be evaluated on par with any other standard codecs then MOS has to be found in a controlled environment. However, these scores can be used to compare different techniques since the samples and environments are same for all algorithms. *An effective VAD algorithm should have high compression and low FLOPS while maintaining an acceptable perceptual speech quality and low percentage misdetection.*

It is necessary to note that the percentage compression also depends on the speech sample used. If the speech signal were continuous, without any breaks, it would be unreasonable to expect high compression levels. The samples here used a mix of monologues and dialogues. The numbers shown here are useful to compare different techniques and show a general trend.

6.8.2 Results

The figures given in this section are comparisons of the six algorithms with respect to the above indices. Totally five speech samples, dialogue and monologue of male voices, Dialogue and Monologue of female voices and one accented female speech sample were used. The samples are of around three minutes in duration to allow listeners to assess the quality of speech to test and to find an average value of percentage of misdetection and compression.

Subjective speech quality

An average of perceptual speech quality grading of five speech samples by seven listeners is shown in Figure 6.6. Listeners graded on a scale of 1 to 5, with 5 corresponding to excellent perceptual quality of speech. Listeners were asked to compare output speech, after applying the VAD algorithms, with original speech. Original speech

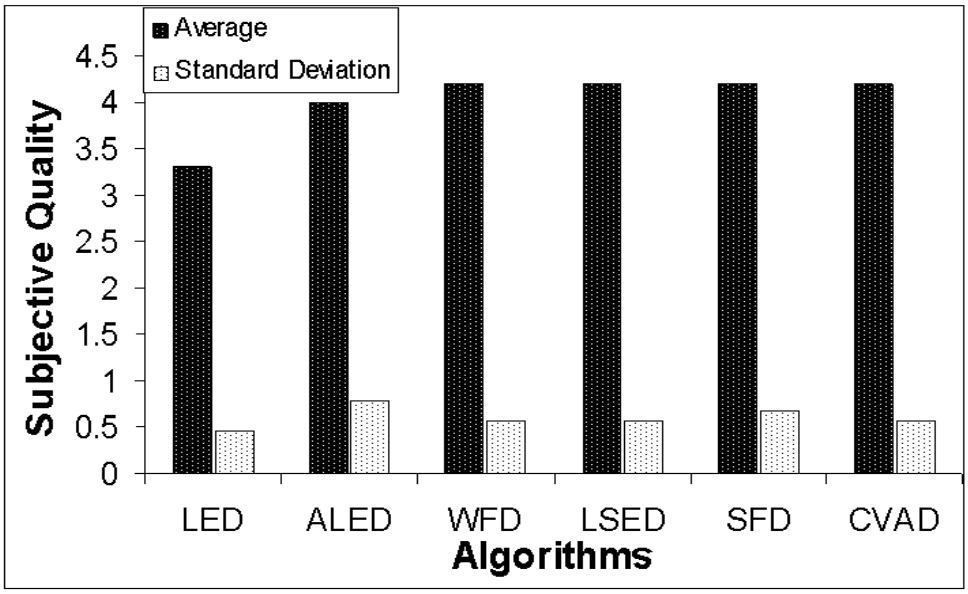


Figure 6.6: **Comparison of subjective speech quality**

(On a scale of 1 to 5; 1 – Poor and 5 – Excellent)

sample is assumed to have grade 5. Listeners were asked to grade on the basis of speaker recognition, intelligibility and frequency of cuts of speech sample. As the speech is not modified, speaker recognition is easy[2] in this case compared to coding techniques where some transformations are used. The perceptual quality depends more on cuts and unintelligibility induced by cuts. Though the numbers here do not mean standard MOS score, the general trend can be observed in the figure. The frequency domain algorithms and algorithm with ZCD produced good perceptual quality of speech compared to time domain energy based algorithms.

Percentage compression

Ratio of number of frames identified as INACTIVE to total number of frames of a sample expressed as a percentage is percentage compression. Higher this value more the compression and gain of an algorithm. Average compression percentage is shown in Figure 6.7. It can be observed from the figure that average compression is nearly 50% for speech samples used in testing.

[2]You mentioned your name as if I should recognize it, but beyond the obvious facts that you are a bachelor, a solicitor, a Freemason, and an asthmatic, I know nothing whatever about you.

- Sir Aurther Conan Doyle, "The Norwood Builder", The memoirs of Sherlock Holmes

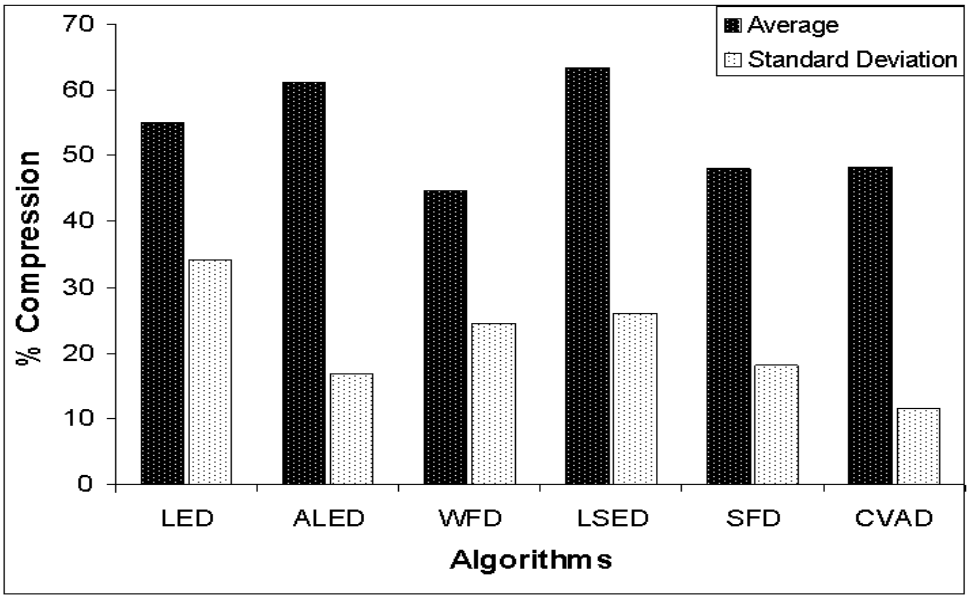


Figure 6.7: **Comparison of percentage of compression**

Percentage misdetection

Although speech quality is subjective and is best judged by the human ear, calculating the percentage of misdetections made by the algorithm is a useful objective test. Lower percentage misdetection leads to better perceptual quality, though not always. A case in point is WFD and ALED. ALED has less misdetection and WFD had more misdetection but the perceptual quality of WFD is better than ALED. Figure 6.8 shows a comparison of percentage misdetection averaged over speech samples. CVAD has less misdetection percentage. Performance of SFD is very close to that of CVAD. As remarked earlier, time domain algorithms perform inadequately.

Requirement of FLOPs

Computational complexity is found using number of FLOPs required. Though these numbers vary depending on the speech samples the deviation is negligible. Figure 6.9 gives FLOPs required for each algorithm. Time domain algorithms out-perform frequency domain algorithms. It can be easily seen that due to DCT computation frequency domain algorithms have higher FLOP compared to time domain algorithms.

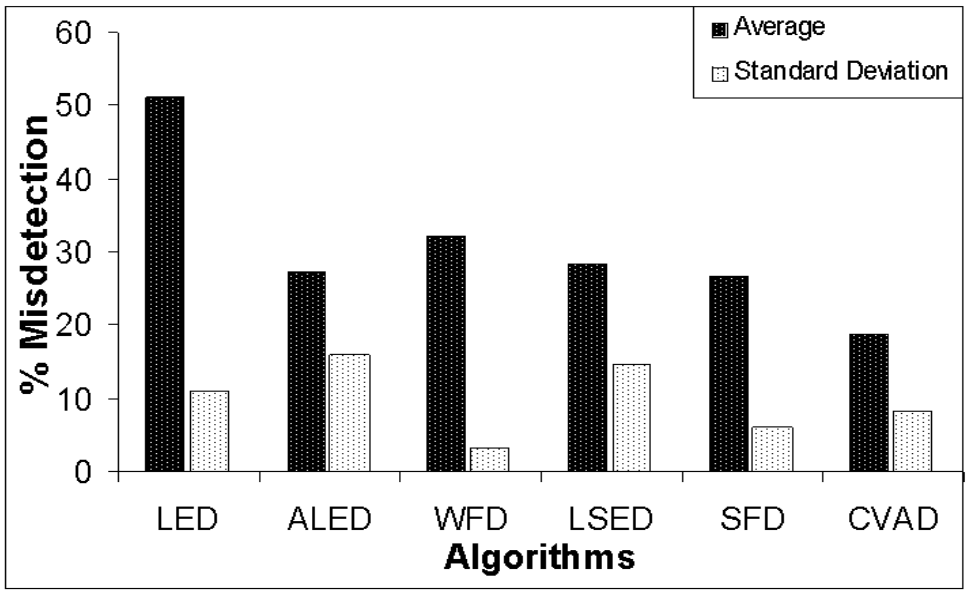


Figure 6.8: **Comparison of percentage misdetection**

Trends

The following are some of the trends observed during the implementation and testing of various VAD algorithms.

(a) The time domain algorithms had the lowest number of FLOPs. This was expected, as the implementation was straightforward.

(b) The percentage compression was low for the perceptual speech quality to be high. This is because some algorithms resulted in high compression rates at the cost of front-end clipping and non-detection of low energy phonemes.

(c) Algorithms based solely on energy did not give acceptable perceptual speech quality with any of the test templates. SFD and ZCD gave better perceptual speech quality.

(d) ZCD was used to recover some low energy phonemes that were rejected by the energy-based detector. However, it also picked up certain INACTIVE frames that satisfied the zero crossing criterion.

(e) SNR affected all the algorithms except the last two, which incorporated SFD. SFD was very effective in speech detection in the case of low SNR.

(f) Approximately, percentage misdetection goes inversely with subjective speech quality.

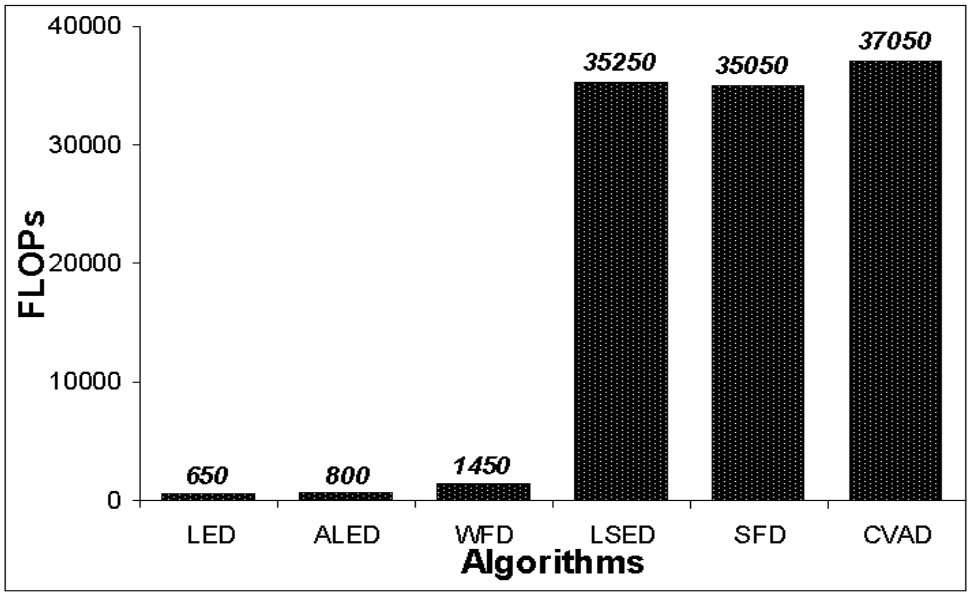


Figure 6.9: **Comparison of required FLOPs**

6.8.3 Comparison with standard codecs

Standard codecs have built-in VAD algorithms. This section deals with the comparison of the proposed VAD algorithms with standard codecs. Two standards ITU-T G.729 (Annex B) [69] and ETSI GSM−AMR [182] are considered here for comparison. G.729 uses Conjugate Structure Algebraic-Code-Excited Linear Prediction (CS-ACELP) to compress speech sampled at 8kHz to 8Kbps. GSM-AMR uses Algebraic CELP (ACELP). Both these include a VAD module. G.729 has a delay of 15ms for 10ms frames whereas for GSM there is no delay except a 20ms coding delay. With packet drops, MOS of the codecs drop drastically. For example, G.729B provides MOS of 3.3 with 5% packet loss.

Table 6.2 gives a comparison. It shows that the computation complexity is less for VAD algorithms proposed in this thesis compared to those in standard codecs. Subjective quality of these algorithms cannot be compared directly with standard codecs as the test samples are not identical. Though percentage misdetection is less in the standard codecs, the cost of computation is very high. The numbers in Table 6.2 are compiled from many sources [203, 204, 205, 206, 207, 208, 209, 210]. MIPS to FLOPs conversion is taken as 1 MIPS = 3.2MFLOPs [211]. GSM VAD assumes stationary background noise. Therefore it misses low energy unvoiced speech [193]. The costs of these codecs that are commercially available are very high; they are often proprietary. VAD algorithms discussed in this chapter have computational ease and attain

Table 6.2: **Comparison of standard codecs and algorithms**

Algorithms/ Codecs	Parameters			
	Subjective Speech Quality	%Misdetection	%Compression	FLOPs[a]
LED	3.3	51.31	55.16	650
ALED	4	27.33	61.1	800
WFD	4.2	32.22	44.56	1450
LSED	4.2	28.48	63.38	35250
SFD	4.2	26.64	48.05	35050
CVAD	4.2	18.67	48.18	37050
G.729B	Note A	13.4	Note B	3.9×10^6
GSM-AMR	Note A	15	Note C	3.4×10^6

NOTE: Values of parameters for standard Codecs are approximate values consolidated from various sources.

Note A: Not pertinent as speech samples are not identical, however, standards quote MOS of 4.

Note B: Data on sample files not available (Maximum rate $8Kbps$).

Note C: Data on sample files not available (Maximum rate $12.2Kbps$).

[a]For G.729 and GSM, MIPS of DSP implementations are converted to FLOPs.

nearly the same level of quality as standard compression schemes. Compression can be enhanced using these algorithms in conjunction with simple ADPCM codecs such as G.726.

6.8.4 Implementation issue

There are two important values of recursive reference adaptation used in all the algorithms wherein E_r may drift towards extremes. First, if speech signal lies only within E_r and kE_r then the reference (in turn threshold E_t) will start building up. In this circumstance if voice signal is low then many speech frames will also be categorised as INACTIVE and it can spiral up as energy of these frames are used again to update the reference pushing it further high. Eventually, it may lead to a condition where in all speech frames would be classified as INACTIVE i.e., the reference E_t is as high as the maximum energy a frame can have. Secondly, if background noise is approaching zero, then all frames will be declared as speech frames resulting in no compression. This condition can occur if the microphone used is unidirectional, and background noise is very low. This condition was observed while using 8 bit quantisation with

less background noise. This in turn makes the adaptation oblivious to noise even if the background noise increases later. Hence E_r remains low throughout. The rate of change of E_r update in the first case is at least $(1-p)+pk$ and in the second case at least $(1-p)$ per frame (using update Equation 6.7).

An assumption here is that during normal operations background noise level is always detectable. However, during implementation and testing of these algorithms there were a few occasions wherein E_r fell and algorithms started passing all the frames. To circumvent convergence of the reference signal to the extreme values, bounds on reference E_r can be used. A limit such as 2% and 25% (depending on k) of the maximum attainable energy of a frame is fixed as a bound for E_r. This problem may be tackled by using the average energy measurement of recent past frames to set an adaptive bound for the reference to operate within.

6.8.5 A remark on comfort noise

An INACTIVE frame is not sent on the network. To avoid the feeling of a 'dead channel' some background noise is generated at the receiver during the silent period. Background noise level is communicated using RTP. RFC 2339 defines the format of such an RTP packet [212]. Since there may be many speakers and channels are mostly occupied, in the conference application, the need for comfort noise generation is less. However, when no speech frame is found for playout, a random signal with amplitude level of 5% of the maximum quantisation value is generated locally at the clients/selectors. It is proposed to send only RTP header since λ is to be communicated. In all the subjective studies comfort noise generation is not used. With comfort noise, the subjective quality may increase marginally.

6.9 Limitations

Two aspects were dealt in this chapter to reduce the requirement of bandwidth for VoIP connections. The first aspect was with respect to multiple streams which are directly concerned with audio conferencing service based on loudness number. Pessimistic and optimistic algorithms use the slow varying nature of loudness number to reduce traffic between selectors. The dynamics of loudness number contributes to the saving in bandwidth. Though it is reasonable to assume that a conference is well behaved and hence bandwidth usage is reduced, both the algorithms do not avoid sudden burst of traffic during floor change or contention. The effect of algorithms is limited to only well behaved period of the conference or when there are only a handful of participants

competing for floors. These algorithms do not provide a constant bound less than $O(d^2)$ all the time as discussed earlier. Another limitation of the proposed schemes is the uniqueness of selection amongst all selectors is lost for a small duration during floor change or competition.

The second aspect covered Voice Activity Detection. There are many methods that are used in speech recognition applications and some of them may not work in realtime. The methods proposed here are applicable for inside buildings, for example, inside an office. These algorithms are not tested with background noise as seen by a mobile set. Though the algorithms are simple to implement, the percentage compression and quality varies with the speech samples. The parameters used here are empirical. Sometimes slightly varying the value of k in classification rule, Equation 6.6 gave better results. Indeed, it is not easy to tune k based on the speech sample. Results are not in standard MOS scores. Saving in bandwidth varies and depends entirely on the sample unlike G.729 or GSM coding schemes. There is still some performance degradation under very low SNR conditions. This can be overcome using cepstral methods [196] which are complex compared to the above algorithms.

6.10 Conclusions

Two classes of heuristic algorithms presented here reduce the traffic efficiently when a few participants are speaking simultaneously. Repairs that are part of conversational analysis explained in Chapter 4 form the basis to expect that only one participant speaks most of the time. Further, conversations are expected to support meaningful interactions. With these ground realities the algorithm should reduce the traffic efficiently. The algorithms are very simple to implement and can be used in place of Algorithm 3.1. For large conferences, it saves considerable bandwidth. Saving in bandwidth is highly dependent on the behaviour of participants of a conference.

VAD algorithms presented here are computationally simple. Participants may select any of the algorithms depending on various features of these algorithms. When participants are not speaking all algorithms will cut traffic completely. The highlight of these algorithms lies in their simplicity to integrate them along with loudness number. Further, they can coexist with codecs such as ITU-T G.726 [70] They are faster, less expensive, and can be implemented easily to achieve similar gains compared to GSM and G.729 codecs. Furthermore, the effect of packet drops on G.729 coding is severe compared to that with plain PCM coding. Thus it is better to use simple coding scheme with some VAD algorithm.

Time domain VAD algorithms are found to be computationally less complex but the perceptual quality of speech is poor compared to frequency domain algorithms. Frequency domain algorithms have better immunity to low SNR compared to time domain algorithms, however they have higher computational complexity. The results consistently show superiority of the comprehensive VAD scheme above all other algorithms. With this scheme good speech detection and noise immunity were observed. The algorithms presented here are found to be suitable for realtime applications and for general environment of VoIP systems with a reasonable perceptual quality of speech within rooms/cubicles where SNR is normally high.

Chapter 7

Experiments, Simulations and Related Analysis

If you make people think they are they're thinking, they'll love you; but if you make them really think, they'll hate you.

- Anonymous

7.1 Introduction

Till now the discussions on the conferencing support involved techniques only at the application layer. It was assumed that the network layer provided the necessary bandwidth. The techniques adopted in reducing the traffic have applied user behaviour and speech analysis. It is necessary and crucial to appreciate the effect of network on realtime packet streams. Hence a study and analysis of network behaviour comprise important ingredients for the VoIP support. This assumes all the more significance when there are no QoS guarantees in IP network. Measurements at the application layer enable fine-tuning of the application without interacting with lower layers. If the network layer provides QoS support it can be additionally leveraged for good quality of speech.

Issues of concern in the study of network behaviour are delay, delay jitter and packet loss. This chapter attempts to discover some of the effects of the other traffic in Internet on VoIP packets by measurements. The measurements are done using a test-bed (covered in Chapter 8). A simple simulation model is used to gain insights into packet behaviour that may be useful for working with real network. There are some proposals to circumvent the effect of network based on these measurements by Haresh [81] using the conferencing implementation[1]. The main aspects that are dealt

[1]Some discussions regarding implementations and design are given in Chapter 8.

with here are the behaviour of the playout buffer on real traces and packet loss in network as well as at the playout buffer.

Delay and delay jitter influence the quality of speech played out at the clients. Discussions on some of the aspects of realtime traffic on Internet are considered first with reference to previous works. Then some of the characteristics found by measurements across Internet using a test-bed are discussed. Later a simple model to find these parameters for the VoIP traffic when it has to compete with general traffic on Internet (best effort network) is discussed.

7.2 Issues in Transmission of Realtime Traffic over IP

The key problems for multimedia delivery over the Internet are the average end-to-end transit times, variations in end-to-end transit times and the loss of packets during transportation. Applications must be able to cope up with these problems and out-of-sequence and duplicated packet deliveries.

7.2.1 Delay

Delay is the time taken by a packet to transit from the transmitter to the receiver. For good quality interactive conferences, the end-to-end delay should be less than 200ms. However, for international calls it may be relaxed to 400ms [32, 78]. End-to-end delay is composed of: (1) the time to sample audio, (2) the time to compress and decompress audio, (3) the network propagation time (including network access), (4) the time spent in router queues, and (5) operating system imposed delays in process scheduling, and receiver buffering. Although delay is not significant over LAN, it is difficult to contain it within 200ms bound over the Internet. Delay depends mainly on the number of hops between nodes and the type of connectivity to Internet. Calculating end-to-end delay is difficult, as it needs time synchronisation. A simple method for calculating transit time is to synchronise the sender and receiver clock. Transit time of packet is the difference between arrival time and transmit time. Accuracy in this scheme is difficult because synchronisation between source and destination in the range of few millisecond is difficult to achieve. Network Time Protocol (NTP, RFC 2030 [213]) is used for synchronisation in Internet. Packet structure of NTP is very rich and supports precision of less than a microsecond. In practice, since today's network and operating systems cannot meet that bound, interactive audio applications cannot be supported

by NTP, neither error-free service is assured.

To overcome this problem, normally end-to-end delay is measured using echo-based techniques. Echo-based techniques suffer because of different properties of the network path in the forward direction and reverse direction. Studies have found that paths through the Internet are often asymmetric i.e. delays measured are not the same when router sequence is not identical in different directions.

Delay significantly influences interactivity of the speech. Reducing coding delay at the application layer and playout delay at the receivers can reduce the end-to-end delay. Further reduction in delay is achieved only if the network provides QoS support.

7.2.2 Delay variation or jitter

Jitter is the variation in delay that successive packet experiences. It is due to the variable time that each packet spends in the service queue of routers. Waiting time in the router queue depends mainly upon the scheduling algorithm and load on router at that instant. In the hypothetical case where delays are same for all the packets this buffer is not required at all. Packets can be played out as and when received. In practice, jitter is a serious problem as it can result in gaps in audio at playout. To avoid gaps in the audio, enough data is buffered in jitter buffer at the receiver before playout. Thus deferred playback evens out at the cost of higher delays seen by a few packets.

This jitter is directly concerned with the inter arrivals of packets and is hence referred to as inter arrival jitter in RFC 1889 [16]. It is defined as the mean deviation (smoothed absolute value) of the difference, D, in packet spacing at the receiver compared to that at the sender for a pair of packets. It is equal to the difference in the "relative transit time" for the two packets. Relative transit time is the difference between a packet's RTP timestamp and the receiver's clock at the time of arrival. For packet i, if S_i is the RTP timestamp and R_i the time of arrival in RTP timestamp, then for two packets i and j, D may be expressed as $D_{i,j} = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i)$.

> **Remark**: To even out this delay jitter, there can be a fixed length playout buffer or an adaptive playout buffer. Fixed length jitter buffer is not advisable as network conditions are dynamically changing. At a selector as streams from various other selectors are received, maintaining an adaptive buffer introduces computational overhead and is difficult to implement. At present, selectors have fixed length buffers as configured during start-up. The focus in this chapter is on the study of the effect of this delay jitter

with respect to the packet size assuming a constant playout buffer at the receiving selector.

7.2.3 Packet loss

If a transmitted IP packet does not reach the intended destination *packet loss* is said to have occurred. Packet loss occurs due to congestion in the network or at the fixed size playout buffer. Realtime traffic can tolerate packet loss to a certain extent at the cost of degraded quality. It is a serious problem in multimedia applications because many codecs rely on state continuity between frames. Packet losses disrupt the state, leading to mistracking at the decoder.

Understand the packet loss process requires an appreciation of why congestion occurs and how it can be avoided. Congestion occurs due to a bottlenecked router in the path. If arrival rate of packets exceeds serving rate capacity of a router, an arriving packet will get queued up in the input queue. If this is full then the newly arrived packet is dropped. This reduces load at the bottleneck router and in turn the router gets a chance to serve the queued up packets. More than 80% of the Internet traffic is TCP traffic. The TCP sliding window flow control algorithm is such that it increases the TCP window size (and thus transmission rate) till a packet loss is encountered. This is done in order to get maximum available bandwidth. When transmission rate increases input queue router gets filled and ultimately over flows. The greediness of TCP connections makes congestion and packet loss unavoidable. Overflow results in routers throwing away some packets. These lost packets are not acknowledged by their respective destination. Then, TCP window size reduces due to lack of acknowledgment and reducing transmission rate. TCP is used for data applications like FTP, Telnet and WWW. These services can survive near zero transmission rate. This is not the case for interactive multimedia applications as they need supply of packets at regular intervals. As congestion is unavoidable, interactive multimedia applications need to survive the congestion period and minimise its impact on quality. A number of alternative approaches to this problem are available and one such method of adaptive packetisation is dealt in [81].

Characteristics of UDP packet loss

Traditional Internet applications depend upon the TCP control algorithm to resolve congestion. There are many implementations of TCP, common among them being Reno BSD 4.3 [81]. In this implementation a sender increases transmission rate additively until a packet is lost; this is taken as an indication of congestion. TCP then decreases

its transmission rate multiplicatively, thus reacting quickly to the inferred congestion. As a result, the level of congestion in the network determines the transmission rate of the TCP sender.

Realtime multimedia applications usually react to network congestion with less flexibility due to their more stringent timing constraints. Packets from a realtime multimedia application must arrive at the receiver before the scheduled playout time; otherwise they are considered lost. Packet losses in realtime multimedia applications, whether due to late arrival or dropped packets, degrade the quality perceived at the receiver. Applications can choose to ignore lost packet or recover from them by retransmission or by receiver based error recovery or masking.

Considerable study has gone [79, 214, 215, 216, 80] into understanding UDP packet loss in the Internet which is normally used for data transfer in realtime multimedia applications. Packet loss and delay are correlated (Section 2.3.6). It is also observed that packet loss increases with the size of packets [214] in Internet. Packet loss due to fixed buffer size at the selectors is considered in the next section.

7.3 Modelling of Playout Buffer

Size of the playout buffer is fixed at the selectors since adaptive playout buffer requires each stream to be tracked and buffer sizes are to be varied accordingly. This introduces computational overhead at the selectors. Selectors consider packets in the queue only at specific time-ticks. Therefore the fixed size playout buffer is modelled as $M/D/1/N$ with constant vacation time equal to service time, where N is the buffer size inclusive of the packet in service. There are many computational methods to find the percentage of packet loss for this scheme. The comprehensive study is provided by Lee [217] which is used as the basis in this analysis. It seems that there are no closed form expressions yet. The derivation here follows from [217] and uses an expression found in [218] for $M/D/1/N$ queue.

Lee [217] defines two probabilities. p_n, probability of number of packets in queue at service completions epochs and q_n, probability of number of packets in queue at vacation completion epochs.

$$p_n = Prob\{\text{Packets in system } = n\}; \quad n = 0, 1, \ldots, N-1 \tag{7.1}$$

$$q_n = Prob\{\text{Packets in system } = n\}; \quad n = 0, 1, \ldots, N \tag{7.2}$$

These are given by

$$p_n = \sum_{k=1}^{n+1} g_{n-k+1}(p_k + q_k); \quad n = 0, 1, \ldots, N-1 \tag{7.3}$$

$$p_{N-1} = \sum_{k=1}^{N-1} g^c_{N-k}(p_k + q_k) + q_N \tag{7.4}$$

$$q_n = h_n(p_0 + q_0); \quad n = 0, 1, \ldots, N \tag{7.5}$$

$$q_N = h^c_N(p_0 + q_0) \tag{7.6}$$

The normalising factor is

$$\sum_{k=1}^{N-1} p_n + \sum_{k=1}^{N} q_n \tag{7.7}$$

$$g^c_k = \sum_{i=k}^{\infty} g_i \text{ and } h^c_k = \sum_{i=k}^{\infty} h_i$$

where, g_i and h_i are the probabilities of i arrivals in a service time and vacation time respectively. Lee proposes a numerical method of solving for p_n and q_n using the ratio

$$\beta_n = \frac{(p_n + q_n)}{(p_0 + q_0)} \tag{7.8}$$

and β_n are calculated recursively [217]. This approach is cumbersome. The new method suggested here uses only departure epochs unlike both service completion and vacation completion epochs as above. Nevertheless, p_n in Lee's approach are found later so that the results can be directly applied to find the percentage of drops and waiting time.

Let p^*_n represent the probability of a system having n packets at a departure epoch. The system is seen only at departures. It is modelled with only this probability; unlike in Lee's approach vacation completion epochs are not considered.

$$p^*_n = Prob\{\text{Packets in system } = n\}; \quad n = 0, 1, \ldots, N-1$$

For this system, arrivals are assumed to be Poisson. Service time is equal to vacation time which is a constant T. Therefore the probabilities of arrivals in both intervals are same. Then probability of n arrivals in T with arrival rate λ (Note: λ is used to denote arrival rate in this chapter. It was used for loudness number in other chapters) is,

$$\alpha_n = \frac{(\lambda T)^n e^{-\lambda T}}{n!} \tag{7.9}$$

Taking $\lambda T = \rho$, an excellent numerical recipe for calculating α_n is

$$\alpha_n = \alpha_{n-1}(\rho/n); \quad \forall n > 0 \text{ with } \alpha_0 = e^{-\rho} \tag{7.10}$$

Whenever a server sees zero packets at a departure it goes on vacations. It returns after constant time T and goes again on vacation if there is no packet. These contiguous multiple vacations till the start of next service time form an idle period. Now define [219],

$$\begin{aligned} f_i &= Prob\{\text{Number of arrivals} = i, \text{ in an idle period }\} \\ &= \sum_{l=0}^{\infty}(\alpha_0)^l\alpha_i = \frac{\alpha_i}{1-\alpha_0}; \quad i = 0, 1, \ldots, N \end{aligned}$$

and

$$f_N = \sum_{l=0}^{\infty}(\alpha_0)^l\alpha_N^c = \frac{\alpha_N^c}{1-\alpha_0} \tag{7.11}$$

$$\text{where, } \alpha_N^c = \sum_{l=N}^{\infty}\alpha_l$$

The transition probability matrix, P, for the playout buffer system seen only at departure epochs is

$$P = \begin{bmatrix} f_1\alpha_0 & \sum_{i=1}^{2} f_i\alpha_{2-i} & \ldots & \sum_{i=1}^{N-1} f_i\alpha_{N-1-i} & \sum_{i=1}^{N} f_i \sum_{j=N-i}^{\infty}\alpha_j \\ \alpha_0 & \alpha_1 & \ldots & \alpha_{N-2} & \sum_{i=N-1}^{\infty}\alpha_i \\ 0 & \alpha_0 & \ldots & \alpha_{N-3} & \sum_{i=N-2}^{\infty}\alpha_i \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & \alpha_0 & 1-\alpha_0 \end{bmatrix}$$

P is an $(N-1)\times(N-1)$ matrix which corresponds to an embedded Markov chain. Stationary distribution or steady state probability π, where $\pi = [p_0^*, p_0^*, \ldots, p_{N-1}^*]$, can be calculated using $\pi P = \pi$. It is examined here whether a closed form solution can be found. For this, $(N-1)$ equations of P are enough. Using P,

$$\begin{aligned} p_0^* &= p_0^* f_1\alpha_0 + p_1^*\alpha_0 \\ \Rightarrow p_1^* &= \frac{p_0^*}{\alpha_0}(1 - f_1\alpha_0) \end{aligned}$$

Define $a_0 = 1$ and

$$a_1 \triangleq \frac{p_1^*}{p_0^*} = (1 - f_1\alpha_0)/\alpha_0$$

Using P again,

$$\begin{aligned} p_1^* &= p_0^*(\sum_{i=1}^{2} f_i\alpha_{2-i}) + \alpha_1 p_1^* + \alpha_0 p_2^* \\ \Rightarrow p_2^* &= \frac{p_0^*}{\alpha_0}(a_1 - a_1\alpha_1 - \sum_{i=1}^{2} f_i\alpha_{2-i}) \end{aligned}$$

Let

$$a_2 \triangleq \frac{p_2^*}{p_0^*} = \frac{1}{\alpha_0}(a_1 - a_1\alpha_1 - a_0\sum_{i=1}^{2} f_i\alpha_{2-i})$$

and

$$a_n \triangleq \frac{p_n^*}{p_0^*} = \frac{1}{\alpha_0}(a_{n-1} - \sum_{j=1}^{n-1} \alpha_j a_{n-j} - a_0 \sum_{j=1}^{n} f_j \alpha_{n-j}); \quad n = 1, 2, \ldots, N-1 \tag{7.12}$$

Remark: The departure epochs of a system without vacations can be found by taking $f_i = 0,\ \forall i \neq 1$ and $f_1 = 1$ since the server is idle only till the first arrival. Then the equation for a_n can be verified with the ones given in [218]:

$$a_n \triangleq \frac{p_n^*}{p_0^*} = \frac{1}{\alpha_0}(a_{n-1} - \sum_{j=1}^{n-1} \alpha_j a_{n-j} - a_0 \alpha_{n-1}); \quad n = 1, 2, \ldots, N-1 \tag{7.13}$$

Brun [218] expresses a_n as $a_n = b_n - b_{n-1}, \forall n > 0$ in Equation 7.13 with $b_0 \triangleq 1$. Using α_n from Equation 7.9. However, a_n are expressed differently below taking into account f_i.

$$\begin{aligned} a_1 &= e^{\rho} - f_1 = e^{\rho} - b_0 f_1 \\ &= b_1 - b_0 f_1, \text{ where } b_1 = e^{\rho} \end{aligned} \tag{7.14}$$

Similarly,

$$\begin{aligned} a_2 &= e^{2\rho} - \rho e^{\rho} - e^{\rho} f_1 - f_2 = b_2 - \sum_{i=1}^{2} f_i b_{2-i} \\ \text{where } b_2 &= e^{2\rho} - \rho e^{\rho} \end{aligned} \tag{7.15}$$

Again by similar manipulations,

$$\begin{aligned} a_3 &= (e^{3\rho} - 2\rho e^{2\rho} + \rho^2 e^{\rho}/2) - (e^{2\rho} f_1 - \rho e^{\rho} f_1) - (e^{\rho} f_2) - f_3 \\ &= b_3 - \sum_{i=1}^{3} f_i b_{3-i} \\ \text{where } b_3 &= (e^{3\rho} - 2\rho e^{2\rho} + \rho^2 e^{\rho}/2) \end{aligned} \tag{7.16}$$

Continuing further, a_n can be expressed as

$$a_n = b_n - \sum_{i=1}^{n} f_i b_{n-i} \tag{7.17}$$

The expression for b_n is taken from [218]. The results of Brun [218] is extended here including f_i which takes care of vacations. The derivation of b_n is omitted here for brevity since the expression for b_n in this case is identical to what is in [218].

$$b_n = \sum_{k=0}^{n} \frac{(-1)^k}{k!}(n-k)^k e^{(n-k)\rho} \rho^k \tag{7.18}$$

Putting $f_i = 0,\ \forall i \neq 1$ and $f_1 = 1$ gives the same values of a_n as in [218] for a system without vacations.

Theorem 7.1 *The state transition probabilities are expressed as* $p_n^* = a_n p_0^*$, *where,*

$$p_0^* = \frac{1}{\sum_{k=0}^{N-1} b_k - \left(\sum_{i=1}^{N-1} f_i \left(\sum_{j=0}^{N-1-i} b_j\right)\right)}$$

Proof: The normalising condition $\sum_{i=0}^{N-1} p_i^* = 1$ implies

$$p_i^*(a_0 + a_1 + \ldots + a_{N-1}) = 1$$

Using Equation 7.17,

$$\begin{aligned} 1/p_0^* &= b_{N-1} - f_1 b_{N-2} - f_2 b_{N-3} - \ldots - f_{N-2} b_1 - f_{N-1} b_0 \\ &+ b_{N-2} - f_1 b_{N-3} - f_2 b_{N-4} - \ldots - f_{N-2} b_0 \\ &\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \\ &+ b_1 - f_1 b_0 \\ &+ b_0 \end{aligned} \tag{7.19}$$

$$\begin{aligned} 1/p_0^* &= \sum_{k=0}^{N-1} b_k - f_1 \sum_{i=0}^{N-2} b_i - f_2 \sum_{i=0}^{N-3} b_i - \ldots - f_{N-1} b_0 \\ &= \sum_{k=0}^{N-1} b_k - \left(\sum_{i=1}^{N-1} f_i \left(\sum_{j=0}^{N-1-i} b_j\right)\right) \end{aligned} \tag{7.20}$$

Therefore,

$$p_0^* = \frac{1}{\sum_{k=0}^{N-1} b_k - \left(\sum_{i=1}^{N-1} f_i \left(\sum_{j=0}^{N-1-i} b_j\right)\right)} \quad \blacksquare \tag{7.21}$$

To find $[p_1^*, \ldots, p_{N-1}^*]$, $a_n = p_n^*/p_0^*$ is used. a_n can be calculated using the Equation 7.12 or 7.17 with b_n.

It is easy to use results from [217] for further calculations of packet loss and waiting times. Equations 7.5 to 7.7, after some manipulations yield

$$\sum_{j=0}^{N-1} p_k = 1 - (p_0 + q_0) \tag{7.22}$$

The solution to the Markov chain gives the probability that the system has n packets at departure epochs. p_0^* gives the probability that a departure leaves no packets in the system, thus starting a vacation. Therefore p_0^* is the probability that a departure starts a vacation. In a vacation period the probability that no packet arrives is given by α_0 since vacation and service times are same in this case. Hence the average number of vacations in an idle period is $1/(1 - \alpha_0)$.

Let V be the vacation period. Now, mean idle time is

$$\frac{E[V]}{(1-\alpha_0)} = \frac{1}{(1-\alpha_0)} \tag{7.23}$$

as T= average vacation time = 1. $\lambda = 1$ ($\Rightarrow \rho = 1$). These values are used for this $M/D/1/N$ queue with vacations modelling playout buffers at selectors.

Theorem 7.2 *Server utilisation, $\rho\prime$, is given by*

$$\rho\prime = \frac{1-\alpha_0}{p_0^* + (1-\alpha_0)}$$

Proof: A system cycle comprises of one idle period and one busy period. Let c_i, $i = 1, 2, \ldots$ be the i^{th} cycle. Let c_i have m_{c_i} vacations. Let the number of departures in a cycle be n_{c_i}, and busy time in c_i be B_{c_i} which is n_{c_i} since the service time is invariant and equal to one time unit. Now server utilisation for k cycles is

$$\begin{aligned}
\rho\prime &= \frac{E[\text{Length of a busy period}]}{E[\text{Length of a busy period}] + E[\text{Length of an idle period}]} \qquad (7.24)\\
\rho\prime &= \lim_{k\to\infty} \frac{B_{c_1} + B_{c_2} + \ldots + B_{c_k}}{(m_{c_1} + m_{c_2} + \ldots + m_{c_k}) + (B_{c_1} + B_{c_2} + \ldots + B_{c_k})}\\
&= \frac{n_{c_1} + n_{c_2} + \ldots + n_{c_k}}{(m_{c_1} + m_{c_2} + \ldots + m_{c_k}) + (n_{c_1} + n_{c_2} + \ldots + n_{c_k})}\\
&= \frac{n_c}{m_c + n_c}
\end{aligned}$$

where n_c and m_c are the mean departures and vacations in a cycle. $1/n_c$ is p_0^* since only one out of n_c departures in a cycle sees the system empty. Therefore

$$\rho\prime = \frac{1}{m_c p_0^* + 1}$$

Substituting $m_c = 1/(1-\alpha_0)$ gives

$$\rho\prime = \frac{1-\alpha_0}{p_0^* + (1-\alpha_0)} \quad \blacksquare \tag{7.25}$$

For Lee's system where p_n refers to service completion epochs, it can be easily seen that $\sum_{k=0}^{N-1} p_k = \rho\prime$. Therefore $(p_0 + q_0) = 1 - \rho\prime$. p_n^* gives the probability that a departure epoch leaves n packets behind when only departure epochs are considered. p_n of Lee's system can be found using p_n^* as $p_n = p_n^*(\sum_{k=0}^{N-1} p_k) = p_n^* \rho\prime$, since both are at the service completion/departure epochs. Probability of packet loss (referred to as blocking probability by Lee), mean queue length, and mean waiting time (including

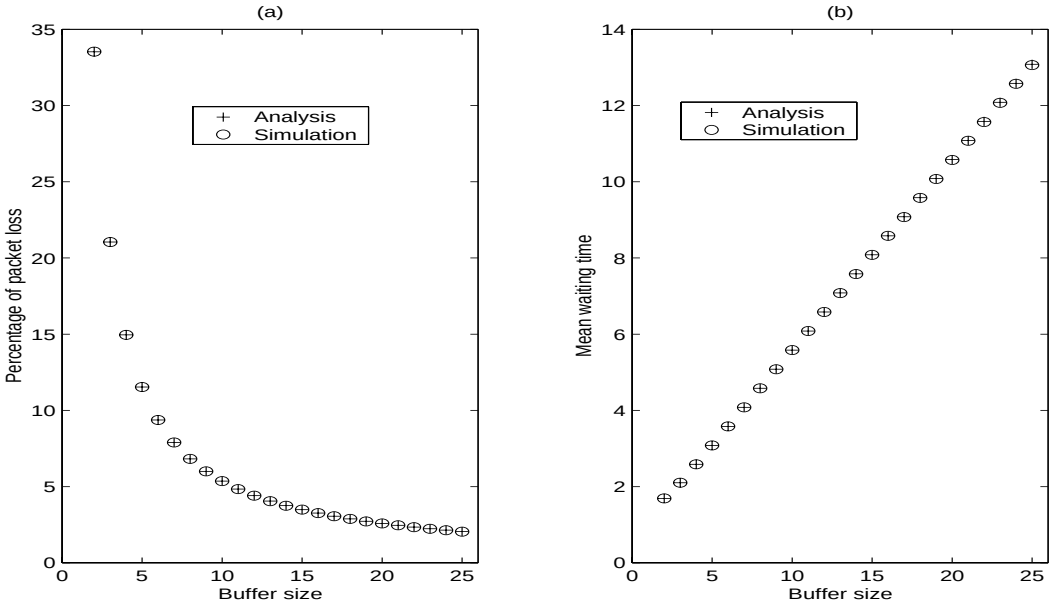


Figure 7.1: **Percentage of packet loss and mean waiting time (analysis and simulation)**

service) are calculated now through expressions given by Lee [217] using p_n.

$$\text{Packet loss probability, } P_{Loss} = (1 - \rho\prime/\rho) = 1 - \rho\prime \tag{7.26}$$

$$\text{Mean queue length, } E[L] = \sum_{n=0}^{N-1} np_n + (1 - \rho\prime)N \tag{7.27}$$

$$\text{and Mean waiting time, } E[W] = \frac{E[L]}{(1 - P_B)} = \frac{E[L]}{(\rho\prime)} \tag{7.28}$$

The advantage of this analysis is in finding the values of p_n easily using p_n^*, which can be expressed in closed form. Simulations shown in Figure 7.1(a) and (b) for $\lambda = 1$ and $T = 1$, with vacation time T equal to service time, agree with the result. Percentage drop reduces with buffer size but increases delay at the buffer, thus adding to the overall ear-to-mouth delay. In fact for deterministic service time, the sample path of occupancy in a larger size buffer is higher than in a smaller one. In practice, percentage of packet loss varies with packetisation interval even though it follows a similar contour.

7.4 Packet Loss for Traces

In this section, the packet loss observed at receivers for real traces across the Internet is presented. The experiment is conducted using the VoIP conferencing test-bed. Buffer

size at the receivers is fixed. For capturing traces of audio traffic, PC-to-PC calls were established from India with remote computers in Switzerland and USA. The remotely located computers were chosen to make audio stream pass through many hops to reflect the Internet characteristics. Traces of audio traffic were captured by running *tcpdump* on a third computer running Linux operating system on the same LAN in India. Experiments with different packet sizes were performed. *tcpdump* log files are processed to find the inter-arrival times. The end terminal logs delay, predicted delay and sequence numbers of the packets.

To record traces, calls were established between the computer *edpc324.cedt.iisc.ernet.in* in Indian Institute of Science and *witness.ee.ethz.ch* in ETHZ, Switzerland and *c91619-a.mntpl.il.home.com* in USA. "traceroute" performed for *witness.ee.ethz.ch* showed that the host is 13 hops away and the USA site was 8 hops away. Traces for different packetisation intervals were collected. Packetisation intervals of 20ms and 40ms are chosen here for comparison. Study on packet loss, consecutively lost packets during transit and delays are presented in [81]. These traces are used to find inter-arrival times of packets at the receiver. Inter-arrival times of these traces are sent to buffers of different sizes. Histogram of inter-arrival time of one trace with packetisation interval of 40ms is shown in Figure 7.2(a). It shows that packets are densely populated around 40ms in this case with peak number of packets at approximately 39.8ms.

Average percentage of packet loss for different buffer sizes and for different packetisation intervals are shown Figure 7.2(b). Packets are dropped at the playout buffer because some packets arrive with inter-arrival time less than the inter-arrival time at the sender (bars to the left of peak shown in histogram). If packetisation delay is more, then the drop at the playout buffer is less compared to the case of smaller intervals. The buffer size is same for different traces and is expressed in number of packets. The network induced delay distribution is more or less same for all packets. If the packets are spaced at larger intervals, then the deviation caused by network does not affect them much compared to packets generated at smaller packetisation intervals. For smaller packetisation intervals the delay jitter induced by the network is higher when compared with its mean inter-arrivals. This causes the packets to form bunches and therefore some of them are dropped at the playout buffer. This is an indirect way of inferring about the delay jitter, which is more for packets transmitted at smaller intervals. Therefore, if some adaptive playout is used at the receivers then for packets with smaller packetisation intervals, tuning has to be more robust to accommodate more jitter.

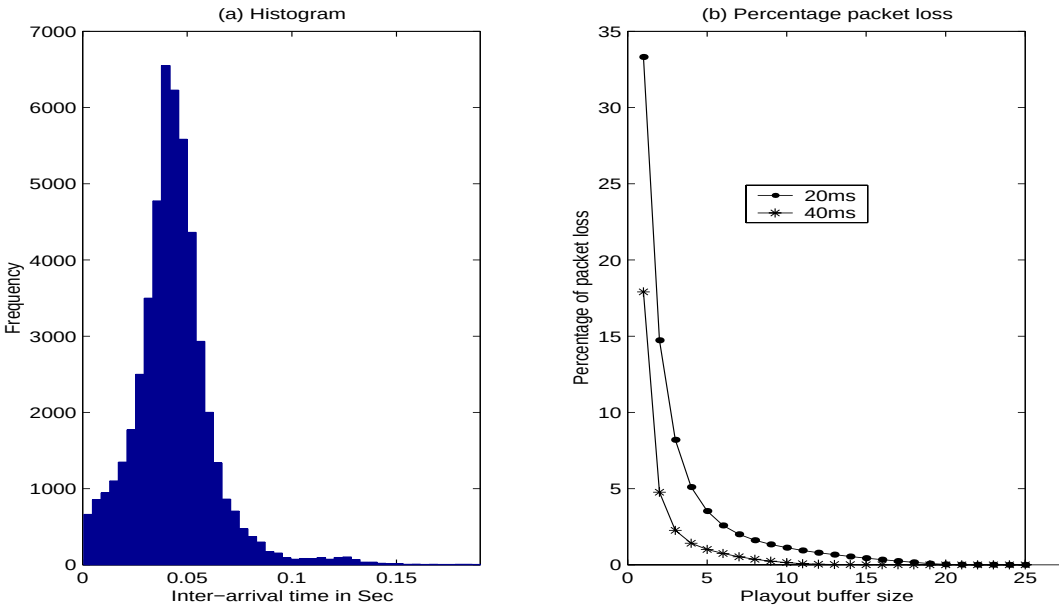


Figure 7.2: **Histogram of inter-arrival delay and percentage of packet loss at playout buffer for traces collected on test-bed**

Here, packet loss at a fixed size playout buffer is used to indirectly infer the delay jitter. One may wonder that observed lower packet loss for larger packetisation intervals may also be due to less number of packets arriving at the receiver. After all, earlier studies [214] showed that large packets experience more drop in the network and hence less arrivals at the receiver. When packetisation interval increases the packet size also increases. Therefore the buffer would have been starving often in case of larger packetisation intervals due to more drops experienced by larger packet sizes. This leads to the study of a simulation model of a bottlenecked link representing network and a playout buffer together. Percentage of packet loss with respect to various background traffic loads is found in this case.

7.5 Simulation

For comparing the net effect of packetisation interval on packet loss including the network induced drop, a simulation model is proposed as shown in Figure 7.3. The model gives flexibility for varying the link occupancy. A simplistic model of the bottleneck router is considered in the model since network drop usually occurs at this router due to buffer overflow (Section 7.2.3). In simulation the background traffic (BT) trace is kept same for 20ms and 40ms packetisation intervals to enable comparison. Background traffic assumes exponential inter-arrivals. Packet inter-arrivals have been studied ex-

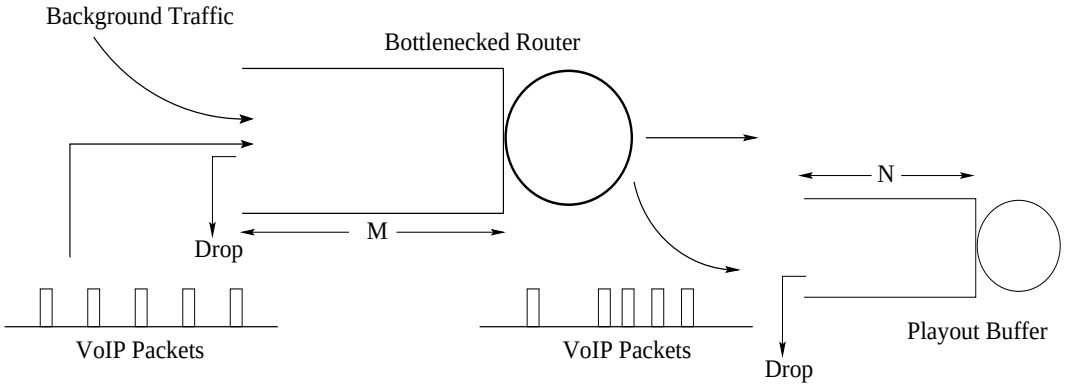


Figure 7.3: **Simulation model**

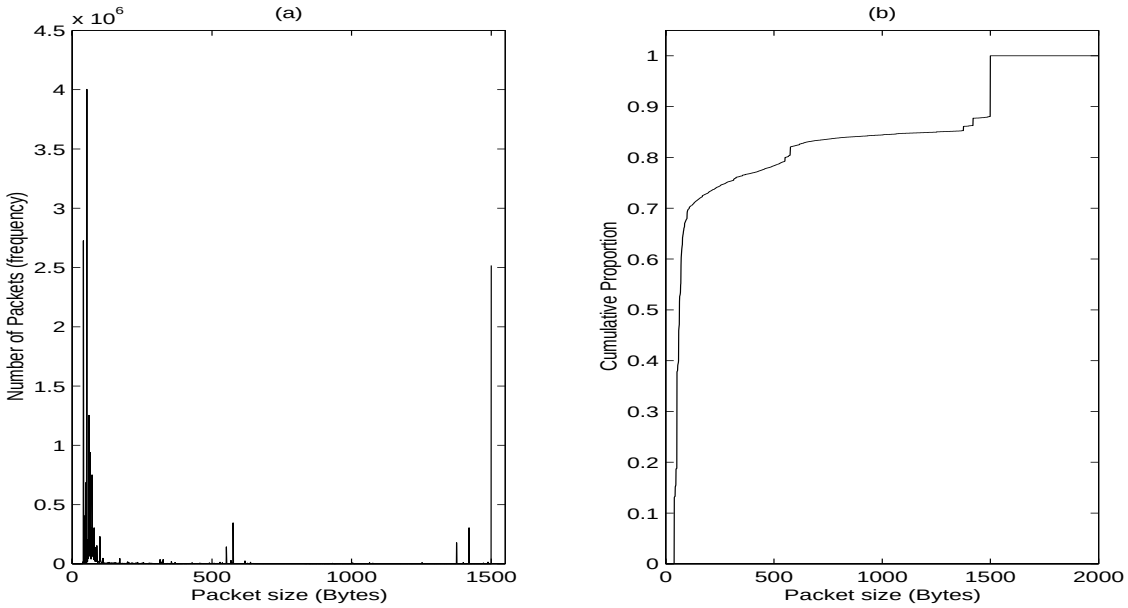


Figure 7.4: **Packet size statistics**

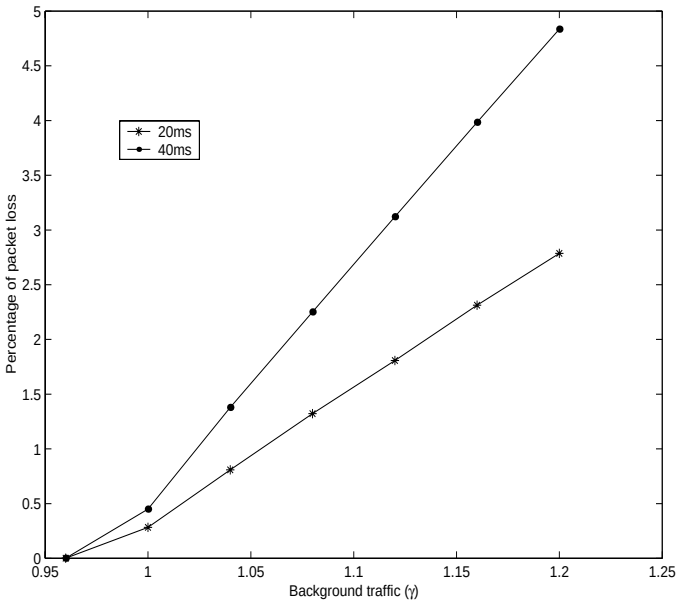


Figure 7.5: **Simulation of percentage of packet loss in the network**

tensively and are approximated as exponentially distributed [216]. Packet sizes are taken from real traces. At the main router of subnet of Indian Institute of Science, Bangalore, seven traces totalling up to 21 million packets were collected on different days and times. Number of packets versus packet size is shown in Figure 7.4(a). A few packets exceeded 1500 Bytes. Cumulative proportion of packets is shown in Figure 7.4(b). It matches with National Laboratory for Applied Network Research (NLANR) and other studies [220]. This empirical CDF is used to generate packet sizes for BT.

The link speed is constant and by varying the arrival rate with mean packet size derived from this CDF one can find total Bytes per sec of BT. Various BT with respect to arrival rate is identified through the factor γ that represents the total input Bytes compared with the link speed in Bytes, which is given by

$$\gamma = \frac{(\text{arrival rate of packets per sec}) \times (\text{mean packet size in Bytes})}{\text{Link speed in Bytes per sec}}$$

The system allows $\gamma \geq 1$ since the queue size is fixed. γ is varied from 0.94 to 1.2. For $\gamma < 1$ the network is not loaded. With $\gamma > 1$ the packet loss in the network is high and hence supporting VoIP applications may be difficult without QoS guarantee from the network. Therefore, γ around 1 is of interest. Average percentage of packet loss for 20ms and 40ms packetisation intervals, with respective packet sizes, is found for 20 simulation runs, each run lasting for 10^7 packets. At higher γ, 20ms packets experience less drop in the network compared to 40ms packets (Figure7.5). The difference between total percentage of packet loss for 20ms and 40ms packetisation intervals are tabulated

Table 7.1: **Comparison of total % packet loss for 20ms and 40ms packetisation intervals**

Queue size	Percentage of packet loss for 20ms packetisation interval in excess of that of 40ms				
	γ				
	0.96	1.0	1.04	1.08	1.12
2	3.1737	2.8265	0.2327	0	0
3	1.8905	1.4942	0.6160	0.0133	0
4	1.1615	0.8147	0	0	0
5	0.8014	0.4865	0	0	0
6	0.5726	0.2896	0	0	0
7	0.4177	0.1604	0	0	0
8	0.3095	0.0688	0	0	0
9	0.2302	0.0052	0	0	0
10	0.1701	0	0	0	0

for different queue sizes in Table 7.1. In the table a 'zero' means 20ms packetisation interval is better (less drop) than 40ms. A positive value shows the amount by which packet loss for 20ms packetisation interval exceeds its 40ms counterpart. For lower γ (less background traffic) and with lower buffer size, 40ms packetisation interval is better. For higher playout buffer sizes, 20ms is always better since packet loss in the network becomes significantly less. This difference increases with the difference between the packetisation intervals. It is felt at this point that larger sized packets (in turn larger packetisation interval) may be compressed to avoid drop in the network since the packet size becomes less. At the same time it causes less drop at playout buffer since packetisation interval is more. This needs to be verified on real network using the test-bed implementations.

7.6 MOS versus Delay and Packet Loss

This section gives a case study of MOS, which is dependent on delay and packet loss, using G.711 codec used in VoIP systems. The E-Model defined in ITU-T Rec. G.107 is an analytic model of voice quality. The basic result of E-model is the calculation of R-factor [221] which is used to find MOS as

$$\begin{aligned} MOS &= 1 \text{ for } R < 0 \\ MOS &= 4.5 : \text{ for } R > 100 \\ MOS &= (1 + 0.035R) + 7 \times 10^{-6} R(R-60)(100-R) \text{ for } 0 \leq R \leq 100 \end{aligned} \quad (7.29)$$

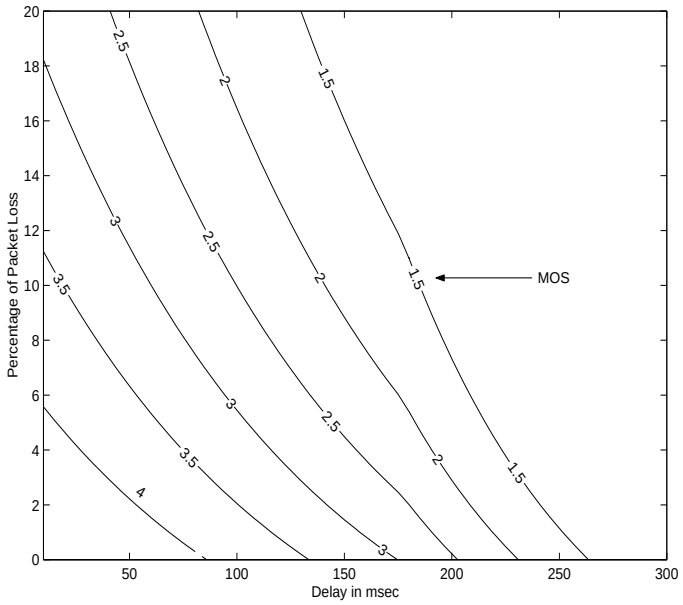


Figure 7.6: **Delay versus packet loss for different MOS**

For G.711 codec R-factor is computed as [221]

$$R \sim 94.2 - I_{d_{me}} - I_{ef} \tag{7.30}$$

Impairment associated with mouth to ear delay

$$I_{d_{me}} = 0.024 d_{me} + 0.11(d_{me} - 177.3)H(d_{me} - 177.3) \tag{7.31}$$

Impairment associated with packet loss

$$I_{ef} = 11 + 40\ln(1 + 10P_d) \tag{7.32}$$

$$R \sim 94.2 - 0.024 d_{me} - 0.11(d_{me} - 177.3)H(d_{me} - 177.3) - 11 - 40\ln(1 + 10P_d) \tag{7.33}$$

$$\text{where, } H(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{else.} \end{cases}$$

d_{me} is mouth to ear delay and P_d is total packet loss. The above expression uses the total delay and total packet loss unlike explicit division of delay in the network, playout buffer and due to codecs as in [221]. Expressions of $I_{d_{me}}$ and I_{ef} given in [221] are directly used here. For different codecs Appendix-I of G.113 gives values of I_{ef} to formulate R-factor.

Figure 7.6 represents the MOS values for different mouth to ear delays and packet losses. Lower packet loss and higher delay also result in lower MOS. For a particular MOS requirement, given the network delay and loss, the balancing act will be with

respect to managing the packet loss and delay at the buffer. Packet loss in network can also be reduced using smaller size packets. In this sense the packetisation interval (in turn packet size) becomes very important to reduce loss in network and delay at playout buffer. As observed earlier coding might reduce packet size and in turn packet loss in network but I_{ef} also changes accordingly. Therefore concealment of packet loss and redundancy become very important aspects. These observations lead to many schemes to reduce packet drops and delay. One such method, called as Adaptive Packetisation with Packet Interleaving, implemented on the VoIP test-bed developed in-house is discussed in [81].

7.7 Limitations

Analysis of the playout buffer model discussed in this chapter is not sensitive to cushioning provided for each talk spurt. The analysis is limited since inter-arrival time is assumed to be exponentially distributed. It helps only in getting a broad picture of the loss and delay at the buffer. The traces showed that distribution of inter-arrival times looks similar to Weibull distribution. The study consisted of only two packetisation intervals. Further studies are needed to clearly find out the effect of different packetisation intervals and coding schemes on the packet loss and delay and, thereby, on MOS. The model of the network shown as a bottlenecked link is too simplistic. The same study on some standard simulator such as *ns*, might give a better picture. The simulation study does not consider adaptive buffer management schemes.

7.8 Conclusions

This chapter extends the model of $M/D/1/N$ system without vacations [218] to $M/D/1/N$ system with vacations. A closed form solution that is simple compared to numerical techniques is proposed. The model of the network, though simplistic, gives an insight into the effect of the network on VoIP traffic. This is useful in understanding the dynamics of VoIP traffic. Simulation model is simple and straight forward. Study of different techniques for reducing delay and packet loss can be carried out easily with these simple simulations. These simulation results match with observed behaviour of the VoIP traffic reported in literature. Though indirect, bringing out the effect of packetisation interval affecting the delay jitter is a feature of this simulation study.

Chapter 8

Implementation

8.1 Introduction

A new scheme for VoIP conferencing using a novel architecture, techniques for bandwidth reduction and automatic floor control were discussed in the previous chapters. This chapter discusses realisation[1] of these features along with other VoIP services in a practical system as an example.

A Proof of Concept Setup (PoCS) developed as part of this thesis is explained in this chapter. A proprietary protocol for exchanging control messages that is implemented on the PoCS platform is first described. Implementation of voice stream management at a selector is discussed. A simple mixing algorithm for tele-orchestra type implementations is also given. A high-level design description of the PoCS with specification is addressed. The software design of CP and selectors of the architecture is also covered in this chapter.

An investigation as to how this architecture can be implemented on an available standard was kept open in Chapter 3. Development of the VoIP conferencing system here uses a proprietary protocol. A proposal for porting this architecture to support this conferencing method on SIP is also presented in this chapter. Allocating the clients to selectors in this distributed architecture for effective services is an important implementation issue. This can be viewed also as the familiar facility locator problem. A heuristic algorithm is presented in this chapter for this. Comparison of heuristic solution with standard LP relaxed instant of the problem is also given.

[1] If you have built castles in the air, your work need not be lost; that is where they should be. Now put the foundations under them. - Henry David Thoreau

8.2 Protocol

The architecture (Chapter 3) proposed for conferencing has three components – Call Processor (CP), Selectors and Clients. The message exchange between them is an integral part of the conference setup before handling the audio part of the conference. Signalling (control message exchange) is required between the clients and CP as well as between CP and selectors. Selector-to-selector signalling is not necessary. Figure 3.2 (on page 65) shows the control flow between CP and clients and CP and selectors. Figure 3.3 (on page 66) shows data (voice) flow between clients and selectors and between selectors. The control messages are communicated using TCP stack for reliable transportation whereas voice packets are sent on UDP. Though the client and selector are separately shown in Figures 3.2 and 3.3, they can run on the same computer.

Selectors talk to each other using multicast or unicast depending on the location of the selectors. For example, if a selector is behind a router that is not multicast-enabled, then that selector would talk to all other selectors on unicast. If there were several selectors that reside in a multicast-enabled network then they would talk to each other on multicast. A selector and a client may use unicast or multicast for voice data transportation. As most of the LANs support multicasting, selectors will normally talk to their clients on multicast. However, in case a client cannot receive multicast packets, the corresponding selector has to send a mixed (with preset weights) unicast packet to those clients or even un-mixed streams.

As described in Chapter 3 the CP implements all the control messaging and is responsible for conferences. A client will be invited to a conference by another client, the moderator of the conference, or will join an existing conference, if permitted, through a password. To get into the conference, the client has to contact the CP. If conferencing is allowed for the client, the CP informs all other clients about the arrival of a new client. As and when a client joins the conference, the corresponding selector will be informed by the CP about the arrival of a new client. The selector will then get ready to serve the new client. If the arrival of a new client brings in a new selector, which in turn brings in a new selector group into the conference, then all other existing selectors will be informed by the CP about the selector just arrived. Value of N_{Max} is programmable at CP. The CP communicates N_{Max} to all the selectors. Globally unique selection of the clients whose packets are to be mixed is done using a distributed algorithms given in Chapter 3 and enhanced in Chapter 6.

Session Initiation Protocol (SIP) does not explicitly provide conference support and it assumes that User Agents (UAs) are capable of executing it themselves or with the help of some servers that can understand SIP messages. H.323 does not address the echo

problem and assumes that the mixed stream would not cause echo. Communication between MC and MP in H.323 is not to be standardised.

A proprietary protocol was developed to implement and test the techniques described in previous chapters. This protocol over comes the limitations of SIP and H.323 protocols. However, its structure is very close to that of SIP protocol. Clients and servers developed in-house use this protocol for call/conference setup and maintenance. A useful aspect of the description of the protocol here is showing the requirements of such a setup. The protocol described here can easily be transformed to text-based signalling such as SIP. For the sake of brevity, only a part of control messaging (i.e., CP to selectors) that is needed for selectors to operate (for supporting a conference) is explained here as an example. However, complete suite of protocol for supporting VoIP calls and conference is designed for the implemented solution. The message that is exchanged between CP and clients can be on the same lines using the same control structure.

8.2.1 Control message structure

The protocol for communication between the CP and selector is explained in this section. As mentioned earlier only control messages or packets are exchanged between a selector and the CP. Data (voice) packets are exchanged between clients and between clients and selectors without intervention of CP. Fields that are used in the control packet structure are considered and their use in the conference scenario explained. The control message structure is shown in Figure 8.1.

1. **Message_Type (16 bits)**: This is the message type which conveys the request that is to be executed by the CP or selector. For example, to start a new conference the "start conference" message is sent from the CP to selectors and the control message contains START_CONF type in the Message_Type field. Some of the other message types are ADD_SELECTOR, REMOVE_SELECTOR, ADD_PARTY, REMOVE_PARTY, END_CONF, MULTICAST_ENABLE, MULTICAST_DISABLE and NO_OF_CLIENTS_IN_CONF. Selectors send SELECTOR_LOGIN during the start-up (Examples of messages from clients to CP are CALL_REQUEST, LOGIN, HOLD_CONF, CONF_ACCEPT, LEAVE_CONF, etc.). These messages are self-explanatory. CP in the beginning sends the GLOBAL_NMAX message that contains $N_{Max}$to be used at selectors when selectors come up and register through REGISTER_TO_CP.

2. **Signals_for_Future (16 bits)**: This essentially expands the Message_Type field.

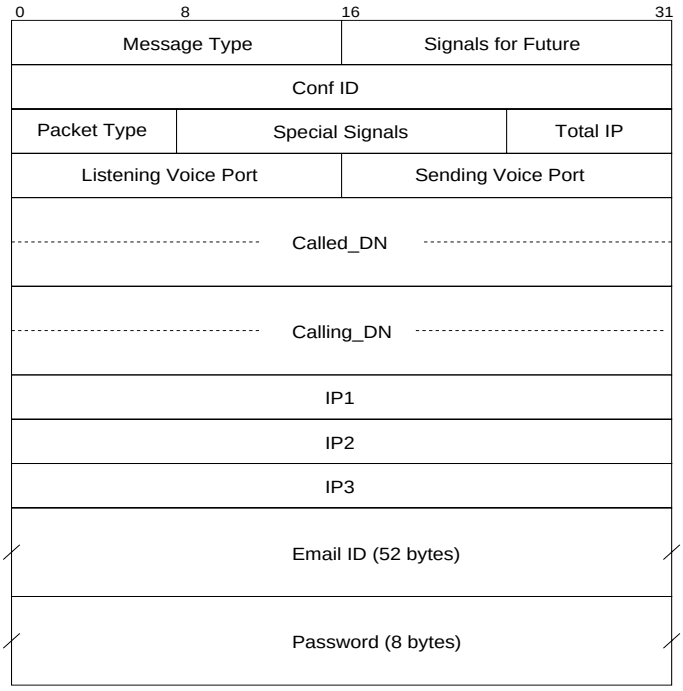


Figure 8.1: **Control message structure**

As Message_Type field may not be sufficient when new facilities are added to the existing facilities. There may be a need for some Signals for future expansion. Also, the message structure may be changed using this field. That is depending on the information in this field further fields may be interpreted differently. a different protocol may be embedded into this message type at a later stage.

3. **Conf ID (32 bits)**: Selectors will be handling many conferences at the same time. This field is used to differentiate between them. It indicates the conference to which the present message applies.

4. **Packet_Type (8 bits)**: It is for use when many standards coding schemes have to be supported and interoperability is to be handled. This field is not used at present.

5. **Special_Signals (16 bits)**: This field contains special signals along with a control message. For example, along with the number of parties in a conference message, one can send an extra information to indicate that an additional party has arrived into the conference. This field can be for new facilities to be implemented without changing the message structure drastically. This field is used along with the existing messages to send more information as and when needed.

6. **Total_IP (8 bits)**: This field contains the number of IP addresses that are sent along with the present control message. This extra information will help in retrieving the IP numbers sent in the control messages. For example, when a new conference is started, multicast Group IP addresses for selector-selector and selector-client communication are to be sent along with the client's IP number.

7. **Listening_Voice_Port (16 bits)**: This port is used to convey to a selector the port on which data packets from clients are to be received.

8. **Sending_Voice_Port (16 bits)**: This port is used to convey to a selector the port on which data packets are to be sent to clients. In case of a group (multicast) address, the port along with the IP number in IP number field forms a unique address for the clients in a particular multicast group.

9. **Called_DN & Caller_DN (64 bits each)**: When telephone terminals are allowed in a VoIP conference through media Gateways, there should be a way of identifying those hard telephone terminals as they do not have IP numbers. Therefore, the clients are identified using a globally unique Directory Number (DN). Called_DN and Caller_DN are 64 bits numbers that are used in identifying the Directory Number of the called and calling parties. To facilitate a globally unique number, the length of this field is fixed at 64 bits.

10. **IP Numbers 1,2 &3 (32 bits)**: IP number fields hold IP addresses, which carry the IP number of the client joining the conference and the multicast group IP numbers for selector-selector communication and selector-client communication.

11. **Email_ID (52 bytes)**: This field contains email address of a client. Clients can be identified using their email IDs so that dialling a number can be avoided. Email ID enables a user to log in from any terminal and receive calls without having a tight coupling between the user and the client terminal.

12. **Password (8 bytes)**: This field is used for authentication if required.

8.2.2 An example of a conference

A ladder diagram for building a conference with three clients supported by two selectors is as shown in Figure 8.2. Any standard [91, 92, 93] can be used for messaging between clients and the CP. For testing on PoCS, the message structure explained in Section 8.2.1 is used. Message structures and values in different fields of messages are not discussed here explicitly. For brevity, only messages between CP and selectors and CP

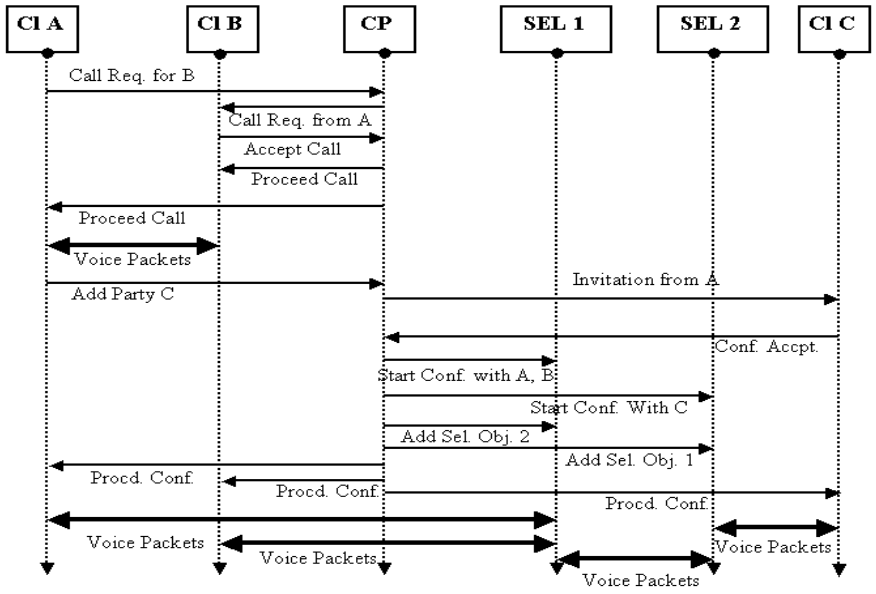


Figure 8.2: **Ladder diagram of a conference**

and clients are discussed here from the point of view of starting a conference. Selectors during start-up are required to connect to the CP by registering with the CP. The CP will convey N_{Max} to the selectors when they contact. Two clients can call each other and can have a point-to-point call without selectors. When a third party is invited to join the present call the existing call will be upgraded to a multiparty call or conference. A selector selects audio packets from active clients and mixes it, if required by a client, and sends it to its clients for playout. When a third party is added to the existing one-to-one call, the CP will assign selector(s) to all the three clients based on network parameters and entities such as number of hops between selectors and clients, the routers between the selector, the location of clients, and load on selectors. The CP will also select the group multicast addresses for selectors to communicate with each other and selectors to send multicast audio packets to their clients in their domain. In what follows, it is assumed that clients are already bound to a selector in the network. The steps for setting up a conference are the following.

1. A client already in a One-to-One call requests the CP to add a third party to upgrade the call to a conference. In case of a booked conference, CP will start a conference at the starting time of that booked conference and CP does not wait for clients to start a call first and then upgrading it to a conference. In this case clients themselves send JOIN messages to CP with Conf_ID and password to join the conference.

2. The CP will contact the third client and if the third party agrees, the CP will proceed to build the conference.

3. The CP sends START_CONF message to the selectors corresponding to the clients that are in the conference along with the multicast addresses as mentioned above.

4. All other selectors which are involved in supporting this conference will be sent ADD_SELECTOR message with selector's IP. The new selector will also get this message for adding all other selectors. This enables the selectors to build a database for each other so that the audio packets from each selector can be stored for further processing.

5. Then the CP sends PROCEED_CONF message along with the multicast address on which clients can listen to audio packets from the selector.

6. The CP sends REMOVE_PARTY message to the corresponding selector when a client leaves the conference using LEAVE_CONF message. A moderator may remove a party by sending a message REMOVE_PARTY to CP with the party's email address.

7. The NUMBER_OF_CLIENTS_IN_CONF message is sent to all other selectors to take necessary steps.

8. END_CONF message will be sent to the selectors once all clients leave the conference. In case of a booked conference it is sent at the end of the booked conference duration.

Messages and flow of events are shown in Figure 8.2. The format of the audio packets which flow from a client to its selector and from a selector to its clients, is briefly considered here for completeness.

8.2.3 Format of RTP packet

RTP packet format is given in Figure 8.3. RTP is used only for voice transmission. The format shown here is a small extension of RTP [16]. There are a few extra fields proposed to support conference facility with multiple unmixed streams and these are accommodated into the existing RTP packet as a part of the payload. Packets from clients have many fields such as Loudness Number along with the usual RTP fields. In case an RTP packet is big (more than MTU) it may be fragmented into two or three

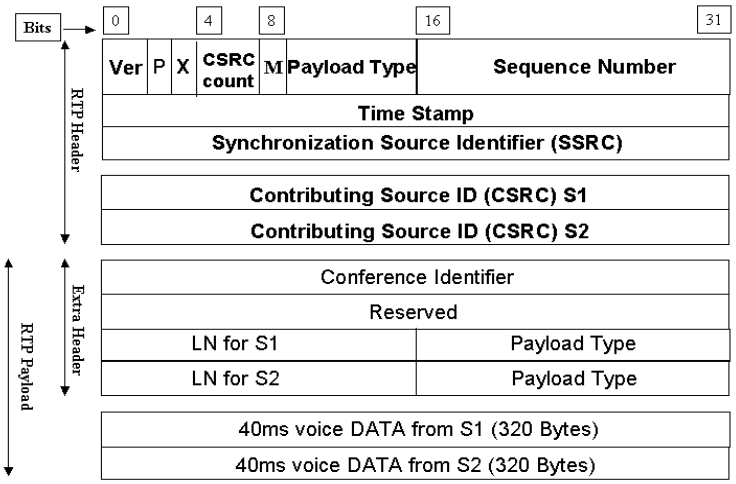


Figure 8.3: **RTP packet structure**

chunks. This is usually needed for exchange of packets between selectors wherein a selector has to send N_{Max} streams for each time slot. The structure of a packet from a selector to its clients with N_{Max}= 2 is shown in Figure 8.3. The packetisation is done every 40ms in this example. Since the coding used is 8 bits PCM, data length is 320 bytes. Loudness Number can be sent using 16 bits. The next 16 bits field, Payload, can be used if different clients in a conference use different form of coding. This will help in decoding speech and also identifying the length of the RTP packet. Currently, on PoCS test-bed, 8 or 16 linear PCM encoding schemes are used at all clients uniformly.

RFC on RTP [16] gives two options for concatenating many media streams. The first option is to use optional header extensions provided by RTP. It may not be possible to use this header for all implementations. The second option provided for specific implementation requirements uses the ALF scheme. Extra fields and information can be put inside RTP payload for specific application needs. This information is provided at the beginning of the payload.

Figure 8.3 also shows standard RTP header along with application specific payload. At present on PoCS, SSRC and CSRC are all IP numbers to identify a selector/clients sending this RTP message and contributing clients who are finally selected. SSRC can be a random number as given in RFC. However, since CP identifies each client with its IP and handles control messages, there is no need for random identifiers other than IP numbers. CSRC field may not be used if a mixed packet is sent and for packets sent by a client to its selector. The extra header that is a part of RTP payload (specific to this implementation) consisting of Conf_ID and a reserved double word is used to identify

the conference since a client can be in different conferences by invoking the application only once. Furthermore, a selector-to-selector data is exchanged using the same format shown, where loudness numbers are to be sent along with packets selected from their respective domains. If a packet is fragmented into two or three for exchanging streams between selectors, then the timestamp will be same for all the fragmented packets though sequence number increments.

It is proposed to use payload type field along with loudness number field to accommodate different coding methods for individual clients. Loudness number is assigned 64 bits at present encompassing 'Payload Type' field. Payload type values are given in RFC on RTP. RTCP [16] is presently not included and it may be used later for feedback, only between selectors.

8.3 Management of Packet Streams at Selectors

When two or more participants in a conference speak simultaneously, all other participants should be able to listen to them concurrently. The streams generated at the same time (with respect to a neutral observer) are to be mixed before playing out at the end terminals. In Telephony this is trivial since lines are physically connected. In VoIP conferencing, this is an important issue since there may not be an ordered continuous flow of voice packets and the delay in packet deliveries is variable.

In voice conferencing, synchronisation to within tens of milliseconds is not a rigid requirement. ITU-T G.114 sets this limit to 400ms [32]; beyond this interactivity is lost. For other applications such as tele-orchestra, musical notes (or audio) should match with each of the performers closely within packet periods. Therefore, there is a need to synchronise each stream continuously before mixing for playing it to listeners. For voice conferencing, these delays are ignored while mixing. With large delays, synchronisation may be difficult. The PoCS test-bed has an option for synchronisation.

Venkat Rangan [222] and Gonzalez [223] have proposed schemes in this direction by giving due consideration for media synchronisation. They do not take clock synchronisation protocols into consideration. Venkat Rangan and Ramanathan [222, 224] propose algorithms based on estimation of generation times to identify which of the packets from various clients to be mixed. The algorithm should be run at each selector before sending packets to clients. However, with the introduction and deployment of Network Time Protocol (NTP [213]) servers and freely licensed applications, synchronisation can be achieved with less computation. Also, secondary servers can be installed and run on the most common Windows or Linux platforms as well. Gonzalez [223]

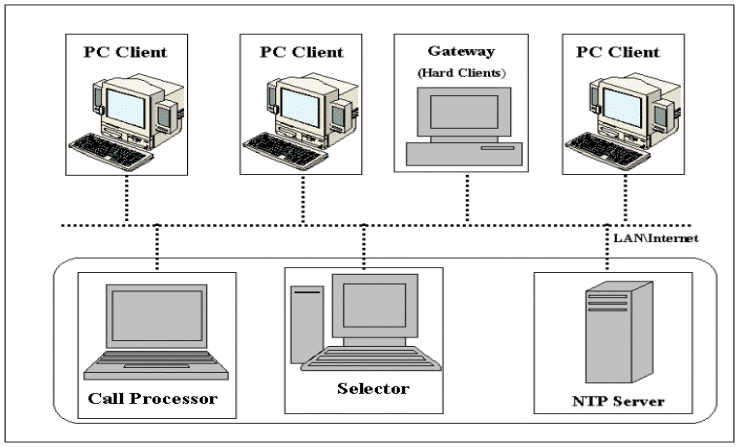


Figure 8.4: **Setup with NTP server**

proposes a new technique that uses the concept of virtual observer. The per-sender approach called differentiated synchronisation is introduced. Also delay estimation algorithms are used.

The algorithm given in this section deals with not only synchronisation of streams before mixing but also management of packet streams at a selector. It includes reception of packets and arranging it in queues before selection. N_{Max} packets are selected from clients whose packets are generated within T ms, where T is the time required to play out/fill in a packet and is fixed. T is the granularity for synchronisation of streams. Along with management of packet streams synchronisation and constraints are also considered here.

In the existing setup one more server called NTP Server is included as shown in Figure 8.4. The NTP Server takes care of synchronising all clients to a secondary standard time server. This NTP program running in the background is responsible for getting the appropriate timestamp for each packet from each client. The resolution of NTP varies from tens of microseconds to tens of nanoseconds on some Unix machines having NTP in their kernels. For VoIP application tens of microseconds would be adequate as the packets contain voice samples of not less than 10ms.

Synchronisation in the presence of NTP is very simple since comparison of timestamps leads to proper synchronisation. However, a list of constraints to be considered while developing the synchronisation algorithm is given first. Some of the constraints may not be required in case of interactive voice streams mixing if the delay induced by the network is within tolerable limits.

8.3.1 Constraints

The following constraints make it difficult for proper mixing of Music or Instrument notes in a conference (tele-orchestra).

1. Packets from various clients may be lost in transmission.
2. Packets from different clients may have different delays. Packets from a client may have variable delays up to a maximum delay jitter limit.
3. Packets that are generated at the same time should be mixed together. Packets that are generated within the time period T of their generation times must be mixed.
4. At the Mixer there is no a-priori information about the delay encountered for packets from a client.
5. There is no information about a client being active in a conference.

8.3.2 Mixing algorithm

The Mixing algorithm must take care of the above constraints. Basically it should find packets in a "Fusion Set". This algorithm also gives the steps for managing audio packets from clients or selectors with or without synchronisation. A fusion set contains packets generated within the packet period T with respect to a global clock. The third party NTP software is more than adequate for this. It runs in the background without affecting the implementation of other components of the solution. Using NTP for synchronisation is advantageous compared to estimation of delays to find packets of a Fusion Set. The proposed algorithm is simple and computationally not intensive. At the same time finding packets in a fusion set is more accurate unlike estimation of respective generation times of packets in a Fusion Set as in [222]. In case a packet is dropped a packet with comfort noise (Section 6.8.5) can be inserted.

On each packet let TS be the Timestamp indicating the time at which the packet is generated. Let D be the maximum delay that a packet can encounter before reaching the mixer. Let the maximum delay jitter be D_j, for all clients Cl_n ($n = 1$ to N). The mixer has separate queues (as shown in Figure 8.5) for packets from each client (and selectors). The length L of the queues (in number of packets it can hold) $Ceil((D + D_j)/T)$, where $Ceil(x)$ represents smallest integer greater than or equal to x. The self-explanatory pseudo-code of the algorithm that is implemented at the selectors is given here.

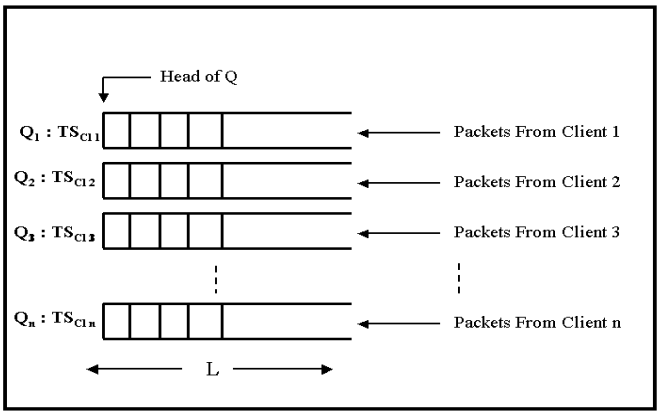


Figure 8.5: **Queues for each streams at a selector**

Step 1. Initialise variables:
Time Stamp $TS_{LastSentPacket} = 0$;
Do {

1. Set Flag $FirstTime_{Cl_n} = TRUE$;
2. Number of packets in queue $K_{Cl_n} = 0$, for each client Cl_n;

} for all N clients (i.e., $n = 1$ to N)
Set the Cushion Number $CN = C$, and $C < L$;

C is based on the delay jitter information about the clients available at the selector. The cushion acts as a buffer if there is a delay for a stream of packets and facilitates continuous play. This can also be estimated with the reception of a few packets in the beginning using one of the many adaptive algorithms such as NLMS.

Step 2. Whenever a packet arrives from a client (say client n) with timestamp TS_{Cl_n},
$If(FirstTime_{Cl_n} == TRUE)$ **{**

1. Enqueue the packet in respective Q_n ;
2. Sort the packets in the Q_n such that Head of Q_n contains
   the packet with least Timestamp ;
3. $K_{Cl_n} = K_{Cl_n} + 1$;
4. $FirstTime_{Cl_n} = FALSE$;

}
$Elseif(TS_{LastSentPacket} < TS_{Cl_n})$ **{**
$if(K_{Cl_n} < L)$
{

1. Enqueue the packet in respective Qn ;
2. Sort the packets in the Q_n such that head of Q_n contains

the packet with least Timestamp ;

3. $K_{Cl_n} = K_{Cl_n} + 1$;

}

}

Else

Drop the packet;

Step 3. To start the Mixer[2] Routine,

Keep checking for every T ms

Do {

$If(K_{Cl_n} \geq CN)$

{

1. Start the Mixer Routine;
2. Stop checking for $K_{Cl_n} \geq CN$;

}

} for all N clients (i.e., $n = 1$ to N)

Step 4. Mixer Routine (if mixer routine is started in Step 3)

For every T ms (using a timer)

While (*if a packet is present* in at least one of the queues**)**

{

A. Find $Min = \min\{TS_{Cl_n}$ of head of queue packets (if present); $n = 1$ to $N\}$;

B. **Do {**

$If(TS_{Cl_n}$ of head of Q packet in queue Q_n $(ifpresent) < Min + T)$

{

1. Select the packet from head of queue for Mixing ;
2. Remove the packet from the queue Q_n to form a Fusion set G and move remaining packets one-step ahead;
3. $K_{Cl_n} = K_{Cl_n} - 1$;
4. $If(K_{Cl_n} == 0)$ $FirstTime_{Cl_n} = TRUE$;

} // End of If loop

} // End of Do loop in step 4B.

C. Mix all top N_{Max} packets based on λ, from G to form a mixed packet P_{Mix};

D. Transmit the packet P_{Mix} to clients on the multicast address. (or transfer the top N_{Max} packets before mixing if mixing is done at the clients) ;

[2]Actually, selector here does not mix the packets, but this algorithm chooses the right set of packets that can be mixed. Mixing is just a scaled linear addition of packets that can also be done at clients.

E. Set $TS_{LastSentPacket} = Min$;
} for all N clients ($n = 1$ to N) // End of while

One important step in this algorithm is the calculation of Min (Step 4A). This brings together all the streams to the level of maximum delayed stream. It automatically queues up all other streams so that the stream delayed most can catch up with other streams. For audio conferences which do not require tight synchronisation, the If statement in Step 4B can be deleted and head-of-queue packets from queues of all clients considered. The assumption about the maximum delay being L packets can be relaxed by the use of linked list for implementing queues. However, realtime nature of the service needs delays needs to be bounded.

For implementation of synchronisation a freely licensed NTP code is used. As synchronisation of the streams from the clients (and selectors if there are many) is needed, only these terminals need to be synchronised. A secondary NTP server serving these terminals would be sufficient. Synchronising to a global NTP server is unnecessary. Also, the refresh period for time correction can be decreased to reduce the load on server. The quality of voice suffers on Win95, as the basic timer supported by OS is not accurate. The quality of voice improved considerably with larger packet size (in turn T). The system works well on WinNT systems with 40ms packet sizes. The whole setup is tested with music as well with a comparable quality of mixed output. On Linux operating systems with a kernel implementation of NTP, synchronisation close to tens of microseconds is possible. Thus the mixing algorithm would be simpler without any adaptive estimation techniques that might introduce error. This algorithm (with 'If' of Step 4B) is used only when tight synchronisation is needed.

8.4 Design of Components of Architecture

In this section requirements of all the services are listed first as Software Requirement Specifications (SRS) before explaining the design of servers.

8.4.1 Software Requirements Specification (SRS)

The description here is intended to be an adequate document giving all the requirements, not a qualified SRS document. The software requirements specifications give the requirements of applications and the setup. The list here does not provide exhaustive descriptions of applications desired; only conferencing part of application is

described. A description of other applications is presented whenever required. The following are the specifications of the PoCS.

Problem statement

To develop a test-bed that supports telephony services such as phone calls, conferences, PBX functions, and Voicemail on IP networks; it should be possible to connect clients to the PSTN to enable an interconnection between the telephony terminals on PCs to PSTN. Application description and functional descriptions are listed below.

Description of application

The application should have the following requirements, facilities and services.

- Components: The application should support soft clients running on PCs and hard clients or handsets, connected to the LAN through media gateways.
- Calls: The application should support one-to-one calls. Any two terminals should be able to connect and have a peer-to-peer voice call. The terminals can be from the pool of soft clients, hard clients or PSTN lines.
- Conferencing: The application should support multiparty conferences of two types,(a) *Invite*, wherein present members invite others and (b) *Join*, wherein a participant can join an ongoing conference with an ID and password. There should be no limit (within reasonable bounds) on the number of parties in a conference. A user should be able to connect to another by dialling a telephone number (digits), Directory Number (DN), or by using email address.
- Compression and encoding: The application should be able to support the G.711 compression scheme.
- Connectivity to PSTN: The application should be able to connect to the existing PSTN exchange, i.e., inter-operability between soft clients, hard clients and PSTN Telephone sets are to be ensured.
- Exchange capability and PBX features: The application should have PSTN-like exchange capability and a few PBX facilities. The PBX features are: Follow me, Call Forward, Call Transfer, Do Not Disturb and Auto-Call back.
- Voice Mail Service: Each client, soft or hard, should be able to send and receive voice mails. The user should be able to send voice mails after editing. Voice mails

can be sent to multiple parties (group mails). Voice mails should be received and sent via FTP (for soft clients) as well as in realtime. Voice mails should have the options to set the delivery time, sent time, sender address/name and subject.

- GUI based User Interface: Soft clients should have GUI based user interface for ease of using. GUI is used for making calls, adding parties to a conference, booking a conference, for voice mails, setting PBX features, Editing, Sending and Receiving voice mails, Login and Logout of the setup, and operations through email addresses.

- Operating Environment: The operating environment for the application is Windows OS and the domain of application should not be restricted to LAN, i.e., the application should support facilities on Internet.

8.4.2 High level design

The architecture

The architecture for conferencing is essentially the one proposed in the Chapter 3. The architecture for the implementation of the application has servers and clients (See Figure 8.6). Servers are Call Processor (CP), selectors (for conferencing), Voice Mail Servers (VMS), media gateways, and PSTN Gateways connected via LAN/WAN. Clients can be soft clients implemented in software running on PCs or hard clients (telephone sets) connected via media gateways. Hard clients are identified from the pre-assigned DN numbers and soft clients are identified by the email address of the user who has logged on at that PC. There can be many selectors, VMS and Gateways connected to one CP. The functionalities of CP and selector are given in the following section.

Call Processor (CP)

CP acts like an exchange in PSTN network. CP will have all the control threads to manage the entire setup. Following are the facilities/services provided by the CP.

1. **Exchange capabilities:** Call switching and setting up calls are done through CP only.

2. **PBX capabilities:** The PBX features are provided by the CP.

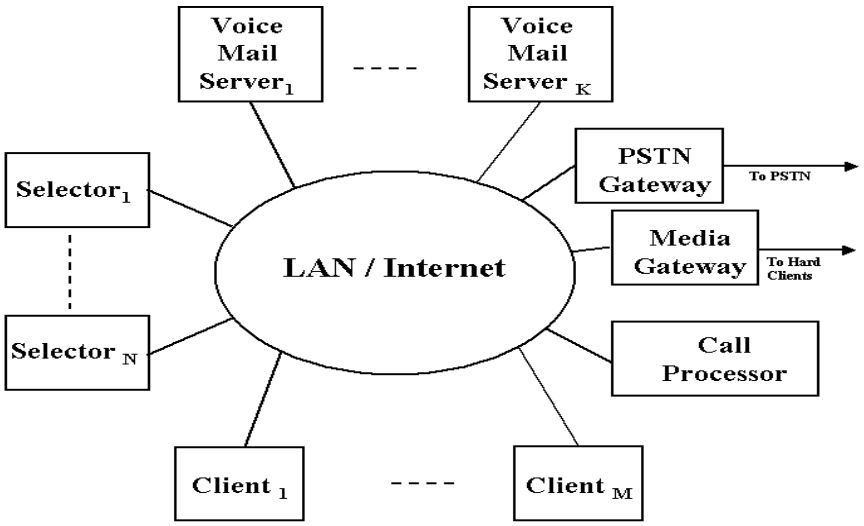


Figure 8.6: **Setup for the VoIP applications (Servers and Clients)**

3. **Conference facility:** CP is responsible for setting up the conference, managing the conference, booking the conference and starting the conference in the case of Join conferences.

4. **Control messages:** CP handles all the control messages in the setup. Control message between any two components in the setup is conveyed through the CP. This design provides a centralized control and helps in call billing.

5. **Maintain a database:** All registered users will be assigned a directory number (DN). Users request for PBX features and data such as email, IP number, etc., are stored in a database that the CP maintains.

6. **Control messages for media gateway:** CP handles the control messages required for connecting to PSTN with PSTN gateways.

Selector

Selector is a server that helps in providing conference facility. It is used only during conferencing. The following are the design highlights and facilities provided by selectors.

1. **Conference support:** Selectors support conference for the soft clients assigned to it by CP.

2. **Selection of Packets:** Selectors select packets from clients in their domain for mixing based on some criteria.

3. **Mixing:** Selectors may mix voice packets selected for dumb clients which cannot mix them.

4. **Selector-to-selector communication:** It connects to other selectors in the case of large number of clients in a conference. Selector-to-selector communication is purely for voice packets (data) and this communication may use multicasting or unicasting.

5. **Connecting to CP:** Selectors are connected to CP throughout their uptime for receiving control messages. Selectors will take commands only from CP for starting and maintaining a conference.

Soft clients[3]

Soft clients run on PCs that interact with users of telephony applications. Following are the features of soft clients.

1. **Provides GUI:** Provides easy access to all facilities supported by the application to a user. Users may use the setup for all the facilities provided by the application only through this GUI of soft clients. Hard clients will be connected through a media gateway. Hard clients use keypad for accessing the services. The GUI also provides facility of editing and interactive playing of voicemails.

2. **Connecting to CP:** Control messaging and service requests are sent only to CP. In case of hard clients the control messages are routed through media gateways. Clients are never connected directly to other servers for control messaging like call request, conference requests and voice mail facilities.

3. **Connecting to selector:** A client connects to a selector for availing conference facility and exchanging only voice packets.

4. **VMS connection:** Clients connect to VMS for exchanging only voice packets or files to avail voice mail facility. In case of soft clients FTP can be used directly for uploading and downloading voicemail files. Connection to VMS is used only when clients are in need of voicemail. A single voicemail can be sent to multiple addressees.

[3]The client software was developed by Haresh and Varchas of CEDT, IISc. See [81].

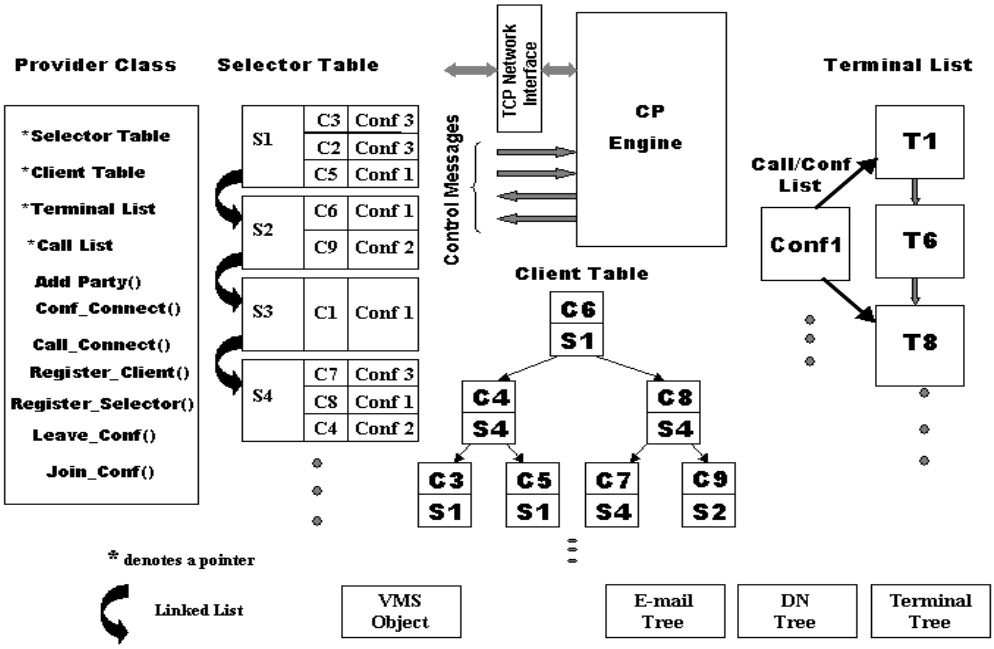


Figure 8.7: **Components of a CP**

5. **Coding:** Clients support G.711 and G.726 encoding format at present.

6. **Multiple calls/conferences:** Soft clients can be logically in multiple conferences and calls at any instant of time and can switch between them freely for voice.

8.4.3 CP design

CP is the centralized control message server. Control messages are exchanged between different components using TCP for reliable transfer of packets. CP does not participate in voice packet exchange. Since CP does not handle voice data, no UDP connections exist in CP. CP must include a network interface module and an exchange engine which handles all service requests from clients.

A schematic of CP is as shown in Figure 8.7. CP maintains a complete centralized database of calls and conferences in progress. It also maintains centralized databases of all selectors and their clients. It is responsible for setting up of call/conference. CP engine invokes the provider object to handle requests from clients.

The network interface block connects to LAN/Internet through TCP. All TCP connections terminate in this block. TCP connections are identified on the basis of IP and port numbers. This is implemented using *Winsock* network support. The data

structures and databases are shown in Figure 8.7. The data structures referred here can also be objects which perform specific operations. Examples of such objects are call objects, selector database objects and terminal objects. The provider object decodes all control messages and calls necessary functions for client requests. It identifies all control messages and maintains a list of terminals of clients and servers.

Clients and servers will have to register with the CP when they become active. These connections last through the life of clients or servers as the case may be. The provider object maintains this login connection in a list called terminal object list. During login, CP builds the mapping between emails, DN and IP numbers.

CP has many databases in the form of height balanced binary search AVL trees for providing the necessary mapping from one form of address and information representation to an other. Using AVL tree reduces search time. For example client table, arranged as a tree structure, as detailed in Figure 8.7, has the mapping from a client to its selector. It contains a database (in the form of AVL tree) of client-selector registration mappings and is updated whenever a client registers for the first time, when a client tries to register to a new selector, or when the CP assigns a selector to a client.

Client Table is another database of selector objects that has information about the selectors registered with the CP, its TCP connection, and the clients that are presently in conference and served by that selector.

An important list of objects that is maintained by CP is terminal list. Each TCP connection (client to CP, selector to CP, VMS to CP and Gateway to CP) is unique. Its details are stored in terminal objects. All transactions on that particular connection are done through terminals in this list. Each call/conference has a different TCP connection for each client. Thus there can be many terminals belonging to the same client if it is in many calls simultaneously.

The other list of objects is call/conference objects. In one-to-one call, the call object has have information about called and calling parties. In a conference call the call object will have a linked list of terminals as in Figure 8.7. There is no separate list in the call object; it is constructed using Conf* pointer in terminal object (see Figure 8.8).

The VMS object of Figure 8.7 has terminals corresponding to clients served with VMS facilities. All VMS related switching and decoding of messages are carried out within in this object. This object builds the communication between a client and the VMS. A PSTN gateway object (not shown in Figure 8.7) connected to PSTN network functions similarly.

Provider class

Provider class is responsible for all functions and service facilitations of CP. It has access points to all databases. All other objects can be accessed through it. Some of the member variable/classes and global variables that are used by it are,

```
* Selector_Table              Tree * CTable
* Terminal_List               TermTree * TerminalTree
* Call_List                   EmailTree * EMTree
* VMS                         Features * FeaturesObj
* Join_Conference             DNtree * DN_Tree
```

A single instantiation of the provider class as a provider object is done in the beginning. It serves as a centralized access point to all tables and databases since it has pointers to those objects instantiated. The provider contains all call and conference management functions. It contains complete information about one-to-one calls and conference calls in progress at any instant. A terminal object is created in the provider object whenever a TCP message is received on CP's listening socket. Each terminal object contains a socket that uniquely identifies a TCP connection with a client.

With the help of linked list of selector objects the provider gives information to the clients about their respective selectors and the conferences they are in. This information is dynamically updated during the CP's session. Multicast address is assigned by the provider object with the help of a database containing multicast address and port number pairs. A set of ten addresses is reserved for selector-to-selector and selector-to-clients multicast. For a conference spanning across selectors, the selectors communicate between themselves on IP_G & P_G and selectors communicate with their clients on IP_R & P_R. The address is allocated in a round robin fashion (wraparound of ten) so that multicast addresses can be reused. These addresses may be taken later from an IGMP server when supported. The addresses for selector-to-selector multicasting and selector-to-client multicasting packets are static due to the non-availability of IGMP in Internet.

The following are some of the important functions handled by the provider object.

```
Login_Client();              Logout_Client();
Register_Selector();         Call_Connect();
Conf_Connect();              Add_Party();
Leave_Conf();                Call_Transfer();
Book_Conference();           Join_Conf();
```

Details on these functions and other members of this class are omitted for brevity. Members of the call class, terminal class and selector class are mentioned here. Class structures are self-explanatory as in the example given below.

```
class Call   \\same object even for conference
{
private:
    int callid;       // Call or Conf_Id
    Connection * conn_ptr;
    Terminal * Orig_ptr;
    Terminal * Ter_ptr;
    BOOL conf;        // Conference flag
    int conf_count;    // Total no. of Participants
    Terminal * conf_head;// Head of Conference List
    Terminal * conf_tail;// Tail of conference List
    unsigned long IPg, IPr;// Group IP
    int Pg, Pgr;     // Group Port
    Terminal * Inviting_Terminal;// Terminal that initiated invite
}

class Terminal
{
private:
   unsigned __int32 IP;  \\ IP number
   int Voice_Port;       \\ Port number
   Call * crptr;         \\ Call/Conference pointer
   SOCKET hSock;         \\ Socket on which the terminal connected
   Next_Terminal * conf; \\ Next terminal if in Conference
   BOOL hold;         \\ Is on Hold?
   Terminal * next;     \\ Next terminal in the list
}

class Selector
{
private:

    unsigned __int32 SelectorIP; \\ IP number
    BOOL Sel_unicast;
               \\ Whether selector is not multicast enabled?
    Client * ClientHead;
```

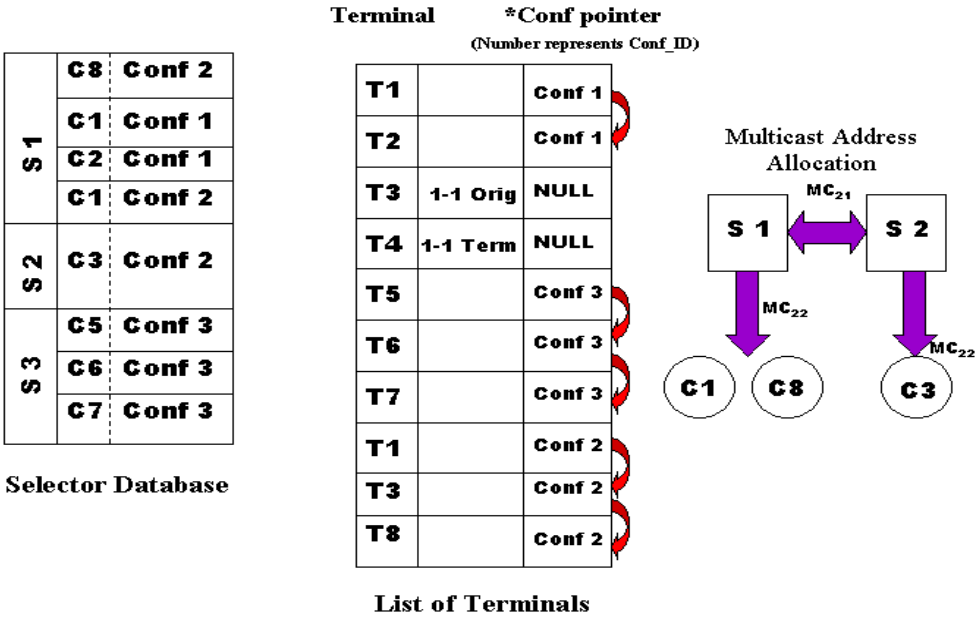


Figure 8.8: **An example of a conference database at CP**

```
            \\ First client in the conference list
    Client * ClientTail;
            \\ Last client in the conference list
    Terminal * SelTerminals;
            \\ To help Provider identify a terminal is a selector
    int ClientCount ;  \\ number of clients served by that selector
    SOCKET SelSocket;  \\ Socket for communication
    Selector * next;  \\ Pointer to next selector in the list
    BOOL SelectorConnected; \\ Is connection to selector Up?
}
```

An example of call and conference structures in CP

Figure 8.8 shows the configuration of databases with seven clients in different calls and conferences. Figure 8.7 shows how the client table arranged in a balanced AVL tree. The S1, S2 and S3 represent the selectors to which the clients are registered. The Figure 8.8 also shows the terminal list, T1 represents the terminal corresponding to Client-1 for Conf-1. When terminals are in multiple calls or conferences multiple terminals are instantiated. For example, T1 is repeated for Conf-2 in the same list. Some terminals are repeated in the list because those clients are in multiple calls or conferences. The link to the right of terminal list shows the manner in which the

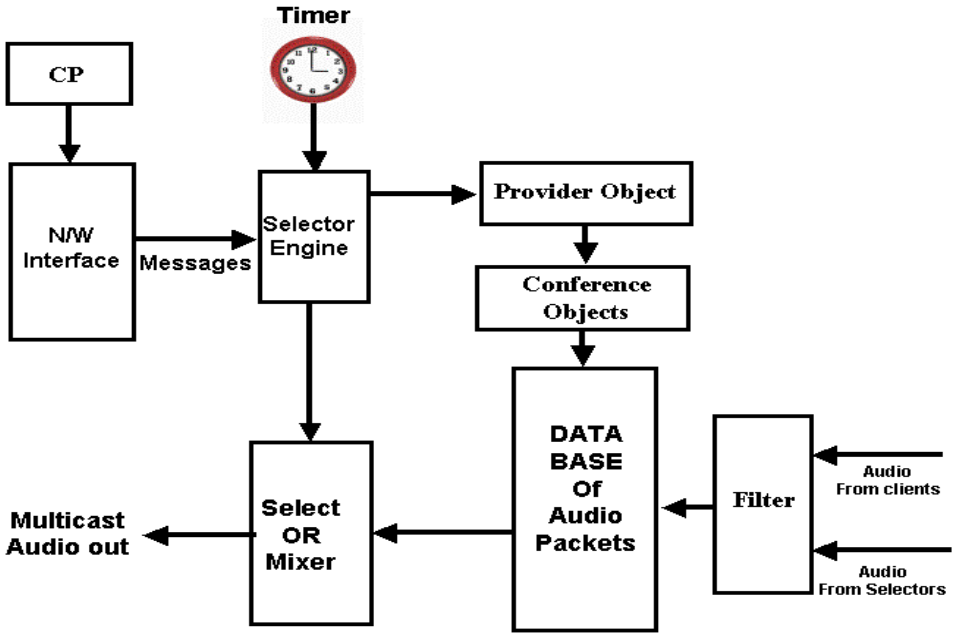


Figure 8.9: **Schematic of a selector**

terminals in a conference are connected.

The figure also shows the selector table where the database of selectors and their corresponding clients are shown. For example Selector-1 has three entries; Client-1 and Client-2 in conference Conf-1, and Client-1 is also in conference Conf-2. Client-3 in conference Conf-2 is maintained in the database of Selector-2. The block to the right represents addresses used by selectors and clients in conference Conf-2. Multicast addresses MC_{21} and MC_{22}, used for selector-selector multicasting and selector-client multicasting respectively are shown.

8.4.4 Selector design

Operations of the selectors are explained in detail in Chapter 3. The packet structure for voice data exchange between selectors, and between selectors and clients is shown in Figure 8.3. The schematic of a selector is as sown in Figure 8.9.

The WinMain program is the selector engine. It also has the global functions. There are two global functions called Select/Mixer and Filter. Select function is called whenever there is a time tick (at intervals of T) using the timer function. Filter function receives audio packets from clients or selectors and puts them in a proper slot in the audio database. A provider object holds all the functions and pointers to all the conferences it is handling. A conference object that resides in the provider object

represents a conference. The provider is the main routine that holds the strings for all the operations of a selector. The provider along with the WinMain program connects to the network and is responsible for all the operations. Selectors never contact clients directly. Selectors connect to CP only. All the information about clients come through the CP. The messages that are sent by CP come to the provider object and it switches to various functions to take the necessary action. Some important functions are:

```
void Create_New_Conference();
void Add_Party();
void Remove_Party();
void Delete_Conference();
void Add_Selector();
void Delete_Selector();
void MulticastEn_Dis();
```

Selectors can serve many clients and can support many conferences (at present it is fixed at 32). The provider instantiates a conference object for each conference and is identified by ConfID. The important objects and functions are summarised below.

Selector engine and provider class: These two blocks are responsible for the proper handling of messages from CP. The provider builds conference objects.

Conference object & database: The conference object and database serves as a hierarchical access to the stored voice data packets from clients and selectors.

Network interface: This is a 'callback' function in WinMain program and is responsible for the network interface for control path messages from CP and data path UDP packets from the clients and other selectors.

Filter: This is a function responsible for sorting the UDP packets and queuing them in the proper order at the client database or selector database.

Select/Mixer: This function performs selection and mixing at every time period Tms. This function identifies N_{Max} packets that are to be selected and sent to other selectors, and globally unique N_{Max} packets for sending to clients in its domain.

Timer routine: A timer is started as soon as the selector is started which gives time-ticks of Tms packet period to be used by Select/Mixer function.

Conference objects and database

The conference object with the help of databases carries out most of the tasks. The Conference object database is shown in Figure 8.10. The conference object has many clients that are listed in another structure called Conf_Terminal forming a linked list of terminals in a conference so that each client's data is filled in that structure. Each

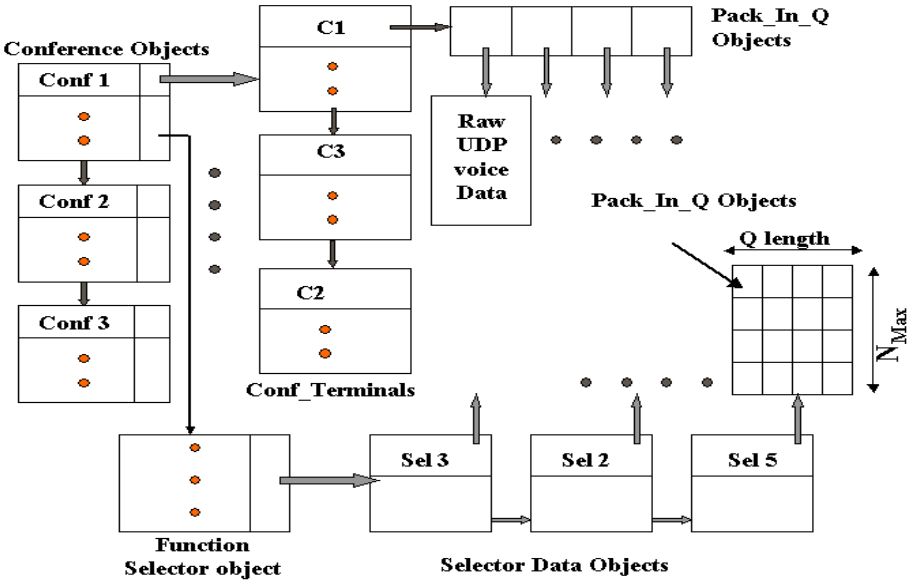


Figure 8.10: **Database structure of a selector**

Conf_Terminal has a pointer to a circular buffer of Pack_in_Q structures that stores the data about each packet that are received from the clients in that conference.

Each conference object has a client bank as a height balanced AVL tree. The key to the tree nodes is IP numbers and the data contains the pointer to the Conf_Terminal entry. The tree structure is used here because it facilitates fast retrieval of the pointer to the locations where voice packets are to be stored. Fast access is required since packets are received from all the clients for all the conferences at each time period T. Since there are many clients in a conference, searching takes more time as the packet queue can be reached only from conference object. Conf_Terminal objects are also linked linearly so that all clients in a conference are accessed easily for extracting the λ of packet of each client.

The conference object has a Function Selector object to store the packets from other selectors that are in that conference. This object is used whenever packets arrive from other selectors. It has selector data objects that store selector specific data; these selector data objects are again stored in an AVL tree for easy access.

Whenever packets arrive at a selector, they are put into the appropriate slot in the queue of selector, i.e., depending on whether the packet is from a selector or a client. This requires a search that uses balanced binary search. The Function Selector class contains all functions to handle database operations for all selectors. Selector objects are also connected to each other as a list so that retrieval at every time tick is fast.

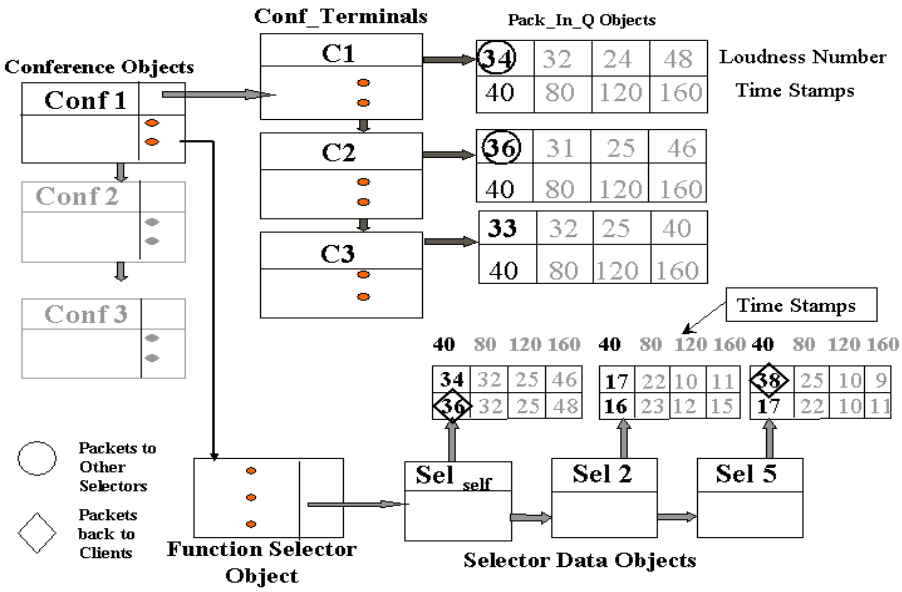


Figure 8.11: **Selector database supporting a conference**

An example of a conference

The conference scenario is shown in Figure 8.11. Consider Conf-1 at Selector-1. This has three clients. Packets from each client are stored according to timestamps at regular intervals of 40ms. The top row indicates loudness number. In this example N_{Max} is taken to be 2. Therefore, only two packets from three clients would be selected at any instant. In the first slot, packets from Client-1 ($\lambda = 34$) and Client-2 ($\lambda = 36$) are selected and sent to all other selectors. These two packets are also stored in the Self-selector database.

The conference object has a pointer to data selector objects that store voice packets from other selectors. In this example, there are three selectors. The other selectors are Selector-2 and Selector-5. Packets from each selector, including itself are queued in the buffers. Each selector database has two queues; totally six packets compete for the top two slots. In the example packets with $\lambda = 38$ and $\lambda = 36$ are selected and would be sent to clients in its domain. Selector screen capture is shown in Figure A.4 of Appendix A.

8.5 A Proposal for SIP Implementation

The SIP standard defined in RFC 3261 [225] and in later extensions [226] is a text-based and HTTP-like signalling protocol. It can establish, modify and terminate multimedia

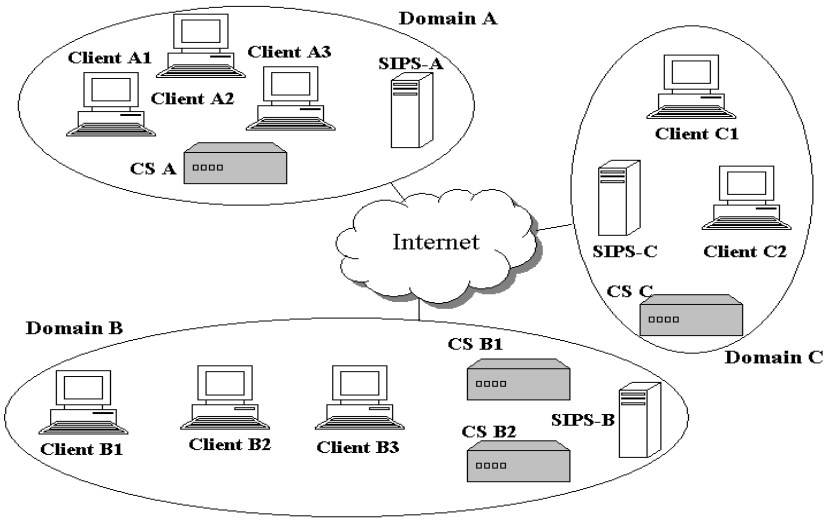


Figure 8.12: **SIP entities in the architecture**

sessions. It does not offer conference control services such as floor control or voting and does not explicitly prescribe how a conference is to be managed. However SIP can be used to initiate a session that may use some conference control protocol such as Simple Conference Control Protocol (SCCP) [135]. The core SIP specification supports many models for conferencing [227, 228]. In server-based models, a server mixes media streams whereas in a server-less conference, mixing is performed at the end systems. SDP [229] can be used to define the media capabilities and provide other information about the conference. More scalable conference control protocols are in SCCP [135]. This section examines whether the proposed architecture implemented using a proprietary protocol can be ported on to SIP standard. More work needs to be done to completely implement all the functionalities described in earlier sections.

The requirement here is to support conferencing for clients in different domains. Three domains are shown in Figure 8.12. A domain can be seen as a LAN or a private corporate network with high-speed data transfer to Internet. Domains can also be a logical group of clients with respective servers equivalent to selector domains (see Chapter 3). This distributed architecture needs distributed control and media handling solutions. The following entities are relevant to the conference service.

8.5.1 Entities

SIP User Agent(UA)

Users' terminals run a SIP User Agent(UA) (equivalent to clients) to connect to servers for availing VoIP services. A UA acts as a front-end for users. There can be an arbitrary

number (with a reasonable limit) of SIP UAs in a domain. Every user is included in one and only one domain at a time. If it moves from one domain to another, it can take advantage of the existing SIP redirection facilities to receive relevant messages to its new location. Implementation of these UAs follows the prescriptions in [225].

SIP Server (SIPS)

A single SIP Server (SIPS) is present in each domain and is entrusted with control aspects of the conferences between one or many clients in its domain. SIPS can be viewed as a physical entity that encompasses different logical roles – SIP Proxy Server, SIP Registrar Server, SIP Redirect Server and SIP Back-to-Back User Agent (B2BUA), which are defined in [225]. This physical implementation enables handling of incoming/outgoing SIP messages by one or the other logical entity according to needs. SIPS can take any logical role at any point of time. The internal logical components of SIPS are described below.

1. **SIP Proxy Server (PS)**: This is an intermediary entity that acts both as a server and a client for the purpose of making requests on behalf of other clients. A PS primarily plays the role of routing or sending requests to another entity that is nearer to the targeted user. A PS can interpret and sometimes rewrite specific parts of a request message before forwarding it.

2. **SIP Registrar Server**: This server accepts REGISTER requests and places the information it receives into the location service for the domain it handles. This task is very useful for the PS when it has to route incoming requests.

3. **SIP Redirect Server**: This is a user agent that directs a client to contact an alternate URI in case the requested target user has moved.

4. **SIP Back-to-Back User Agent (B2BUA)**: It is a logical entity that receives a request and processes it as a User Agent Server (UAS). In order to determine how the request should be answered, it acts as a User Agent Client (UAC) and generates requests. Unlike a PS, it maintains dialog state and must participate in all requests sent on the dialogs it has established. The B2BUA (and therefore the SIPS) needs to be stateful (in the SIP terminology) to remember the state of all current calls/conferences underway in/to/from its local domain. Since it is a concatenation of a UAC and UAS, no explicit definitions are needed for its behaviour, i.e. it can perform the same tasks and handle the same request

messages as a UAC or a UAS. Since B2BUA can perform more functions than a regular PS it will be the core entity for conference related signalling.

For communication needs of the four logical entities in SIPS, many interfaces are devoted to communication with the domain Clients (User Agents). Interfaces are also required to communicate with SIPSs in other domains and with the local Conference Server. Furthermore, objects related to calls or conferences currently under way to/from the domain are stored in the SIPS, so that it has complete information about them (fulfilling the stateful condition necessary for the B2BUA as described above). These conference objects, described in Section 8.4.3, contain relevant conference-level information, a list of all clients taking part in this conference, irrespective of their local or remote location. A list of the SIPSs from other domains involved in that conference and a list of the CS(s) operating in the local domain are also to be included.

SIPS should be entrusted with providing total signalling service. SIPS taking care of all control aspects has many advantages, viz., (a) it works as a centralized entity that can keep track of the activities of the UAs in a call/conference; (b) it does all the switching for providing PBX features; (c) it locates the UAs and invites them to a conference; (d) it can moderate billing if required.

Conference Server (CS)

Conference Server (CS) is a entity that is used only for conferences. A CS is nothing but a selector with the architecture described earlier (Chapter 3).

8.5.2 Making the entities work together

For all the entities to work together and provide the desired conferencing service, they need to communicate with each other and exchange messages. Messages can be divided into two distinct types. (1) SIP control messages which are sent via TCP or UDP (using a timeout in the latter case). (2) RTP voice packets are exchanged between two UAs in the case of a point-to-point call. In a conference the voice packets are exchanged between a client and its CS and between CS's. RTP packets may be sent on unicast or multicast. Conference-level control requirements in different entities are given below. (A) User Agents know the IDs (i.e., the IP addresses) of their local Register and Proxy Servers (that are fixed and do not have to be dynamically communicated to the UA), which are necessary to start a call or conference. The address of a local CS is not known to the UA in advance, as there can be more than one CS in a domain. There is

thus a need to dynamically communicate that address to the UA from the SIPS in its domain whenever a UA joins a conference.

(B) A SIP Server has many details about an ongoing conference. It knows the IDs of all registered clients in a domain and the IDs of the clients involved in a given conference. It is aware of the IP address(es) of the CS(s) handling the media part of the conference(s) in its domain. It also knows the identity of remote SIPS in other domains. Most of the information must be dynamically obtained by the SIPSs as the conference progresses.

(C) A Conference Server knows the IDs (IP numbers in this case) of all the local Clients it must handle. The IP numbers of remote CSs are also known so that a subset of local audio packets in each time slot can be sent to them. As it was designed that CSs would only communicate to their local SIPS for SIP messages, new information (such as addition of a client or addition of a CS to serve this conference in a remote domain) will have to go through SIPS before reaching a given CS. Unlike media packets, it will not go directly between those entities.

For these entities to communicate together and to conform to SIP philosophy, it is assumed that the entities can receive and send SIP messages. This is already the case of the UAs, the PSs and the B2BUAs that have communication abilities, which are defined in the relevant RFC [225]. The only extension needed is to assume that the newly created CSs can send and receive a subset of available SIP requests (i.e., CSs are similar to B2BUA). As the different entities need to communicate many messages between them (e.g. the IP address of a Conference Server, a list of remotely involved clients), a SIP request message pair SUBSCRIBE and NOTIFY is used to convey pre-agreed information. These two messages are described in [226]. It is used for event notification in SIP framework. Entities in the network subscribe to the state of various resources in the other entities. They are notified when those states change. The B2BUA will be the SIPS logical component entrusted with handling all subscriptions/notifications for all entities within SIPS. Therefore, it has a central role in the conferencing environment.

Figure 8.13 shows a ladder diagram showing the SUBSCRIBE/NOTIFY messages pair. In message (1), the CS subscribes to SIPS so that SIPS notifies events to which the CS subscribes. The message body contains a description of the resource to which it subscribes. This first message is acknowledged by the sending of message (2). Then the SIPS immediately sends a NOTIFY message (3) so that the CS knows the present state of the resource. Later, as soon as an event occurs in the subscribed resource, messages (5) and (6) (similar to messages (3) and (4)) are exchanged. A subscription is destroyed when a notifier sends a NOTIFY request with the "Subscription-State" as

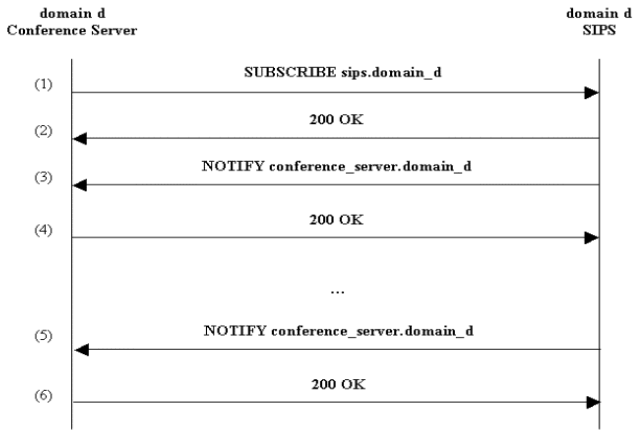


Figure 8.13: **SUBSCRIBE - NOTIFY message exchange example**

"terminated". A subscriber may send a SUBSCRIBE request with an "Expires" header of '0' in order to trigger the sending of such a NOTIFY request. For conferencing with the help of information that every entity holds, one can draw a list of resources a given entity must subscribe to. The subscription dependencies shown in Figure 8.14 are:

1. UAs will subscribe to their local SIPS to be notified to know which CS will handle their audio packets. They may subscribe to their SIPS to be notified of the joining of every other UAs to have the information of those involved in the conference (i.e., their names, location, Source Description (SDES), etc.).

2. A SIPS has a subscription from locally registered UAs so that it is informed of changes in the conference (for e.g. joining, BYE, etc.) making it stateful.

   UAs have to subscribe to the other SIPSs in other domains. This is required to get the information about the ongoing conference. In turn SIPS can provide the local CS(s) with some pertinent information. Examples of information exchanged between SIPS are UAs joining, UAs leaving, and addresses of new CSs that are brought into the conference (to pass it on to CSs of its domain). Thus all the SIPSs must subscribe to each other so that a full mesh of logical connection (for subscribe and notify messages) is built with respect to a conference.

3. CSs will subscribe to local SIPS to be informed about the other CSs present in the conference. It will help in knowing about the UAs and media packets which it will have to handle in a conference.

SUBSCRIBE and NOTIFY messages are used to establish the logical connections between different entities to support a conference. This connection is used to exchange

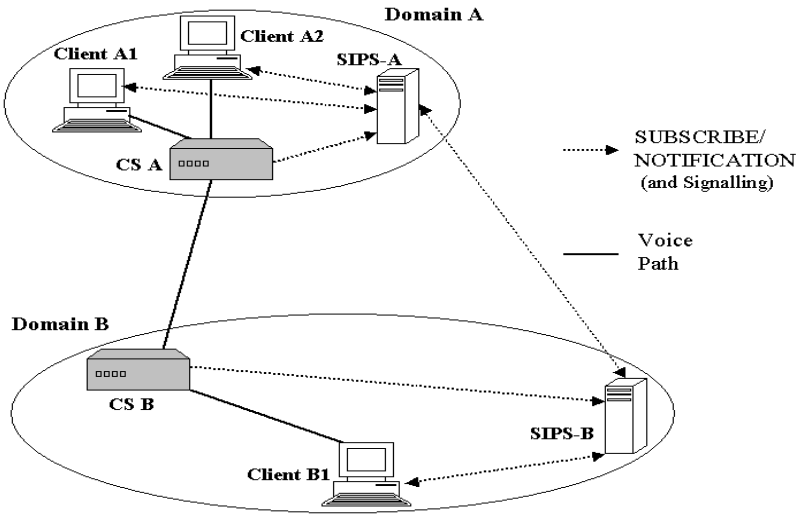


Figure 8.14: **Message flow between different entities.**

the information of the state of a conference when it changes. The next section describes an example of preliminary message exchange for setting up a conference, with two domains.

8.5.3 An example

Two domains are considered in the example shown in Figure 8.13. Establishment of connection for exchange of conference information between them is considered first. Both the domains have a SIP Server (SIPS-A and SIPS-B) and one Conference Server (CS A and CS B). There are two potential SIP Clients in Domain A (Client A1 and Client A2) and one in Domain B (Client B1). After the setup phase, since SIPS, CS and UAs are connected together, usual SIP messages (e.g. INVITE) are used for conference establishment along with SDP [229]. SDP helps in communicating all the necessary information of the conference.

Intra-domain setup phase

First consider intra-domain setup phase in each domain before starting a conference. Message exchange for Domain A is shown on Figure 8.15, there are three different stages:

Stage 1: Registration of CS A to SIPS-A (the B2BUA logical part). After registration, CS A will be informed about UAs joining its domain and about the addresses of other CSs in other domains. The "NOTIFY CS A" request contains no information but is compliant with the SIP specification.

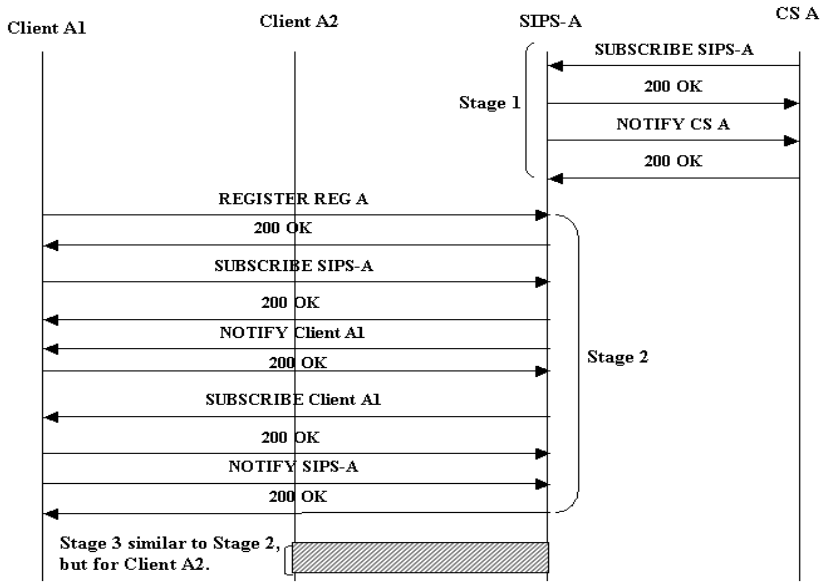


Figure 8.15: **SIP messaging during intra-domain setup phase**

Stage 2 and Stage 3: Each client registers with its local Registrar Server (physically embedded in SIPS-A). Then the clients subscribe to their local SIPS so that they are kept informed of future calls/conferences, and CS handling them. Signalling for one-to-one calls can still go directly between UAs or through Proxy servers without using the elaborate SUBSCRIBE/NOTIFY messaging. In Stage 3, SIPS-A (actually, the B2BUA part embedded in it) registers with the client to be informed about its state in the conference (in particular if it wishes to join/leave a conference), leading to a stateful SIPS. These message exchanges occur for every prospective client. After this initial setup phase, a conference may start at any time inside a given domain. Another phase is necessary so that the conference can span many domains.

Inter-domain setup phase

Before any conference can take place between different domains, an inter-domain setup phase must be realised. It is shown in Figure 8.16. This phase would be contained in a "remote invitation phase" explained below. When the SIPS in Domain A receives third message of the INVITE/200/ACK three-way handshake, used to establish SIP sessions, it is sure that the remote client situated in Domain B will be included in the conference. Thus SIPS-A subscribes to SIPS-B so that it is informed about the events in both domains. In Stage 2, the converse is realised. If more domains are involved in the conference later, SIPS-A will make sure that those other SIPSs already in the conference know about the joining of the new domain so that this inter-domain setup

takes place between all of them. The following are the steps:

- SIPS-A tells SIPS-B about new SIPS(s) that have come into the conference. Then SIPS-B would contact each of them and those would also subscribe to SIPS-B, because of the next point.
- If SIPS-I subscribes to SIPS-J it is reciprocated by SIPS-J subscribing to SIPS-I.
- In a conference where users join themselves (also called an open conference) the SDP would have the initiator domain (say, SIPS-A) to which the new SIPS subscribes and get the information about all the other SIPSs.

  Remark: It is not hard that the starting SIPS should always start these subscriptions. The SIPS of the inviting client may trigger this series of subscription messages.

However, only one inter-domain setup phase can occur between SIPSs in each conference, for after this phase, there is no need to start another setup phase as they are already connected. When all SIPSs in a conference have subscribed to each other, a full mesh of SIPSs is created, which is a guarantee that signalling information is passed between them. In Stage 3, each SIPS notifies its local CS about the ID of the remote CS it has just been notified about. Information about remote clients as and when they join are exchanged between SIPS.

> **Remark:** RTP allows the exchange of participants' information while joining and leaving the conferences and while in the conference. However, this method of keeping the information inside SIPSs in a domain is as helpful as a central repository and can be disseminated *only* when a client asks for it. This avoids frequent exchanges of SDES messages.

Local invitation

Now it is assumed that Client A1 invites Client A2, both in the same domain. This will start the conference. Two clients can interact directly without a CS. However, the example given here is to show how a CS is included. In any case when there are more than two clients a conference will start with a CS. The message flow is shown in Figure 8.17. In the first stage, Client A1 invites Client A2 with a normal SIP invitation message exchange as defined in [225]. The messages go through the local Proxy Server, which is contained in SIPS-A. In the second stage, SIPS-A sends a NOTIFY message

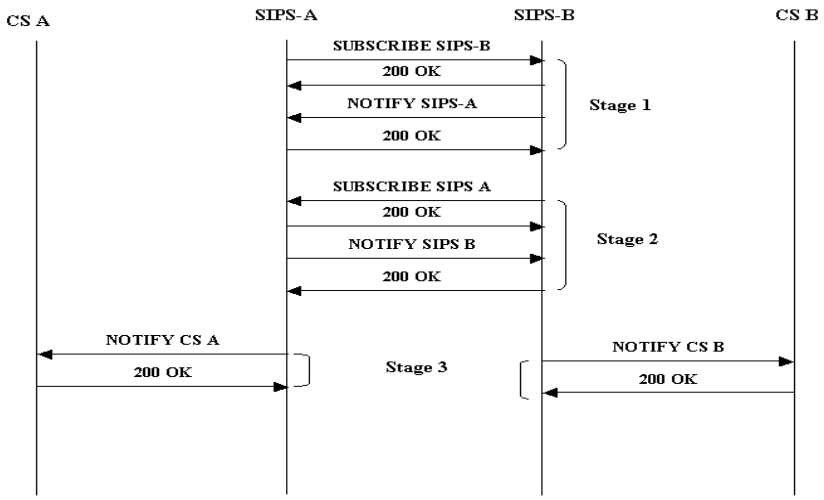


Figure 8.16: **SIP messaging during inter-domain setup phase**

to CS A to inform that two local clients (with their IDs) have joined the conference. Thus, CS A will be able to handle their audio packets. In the third stage, SIPS-A notifies ClientA1 and ClientA2 with the ID of CS A so that they will know which is their Conference Server.

Remote invitation

Remote invitation is used when one of the Clients in Domain A wishes to invite a client from Domain B. Assume that Client A1 invites Client B1 in the conference. Message exchange for remote invitation is shown in Figure 8.18.

In Stage 1, a normal SIP invitation takes place between Client A1 and Client B1. When the ACK message flows through SIPS-A, SIPS A checks whether the inter-domain setup has already taken place. If it has not, then inter-domain setup take place right away as explained before. In Stage 2, SIPS-A notifies SIPS-B about relevant conference-level information (such as the handling CS in the domain and the ID of the involved local clients). Then SIPS-B does the same with SIPS-A. Then, in Stage 3, CS A is notified of the ID of CS B that has just joined the conference. In Stage 4, SIPS-B does the same with its local Conference Server, CS B.

Now that all the entities have been provided with the necessary information the conference starts. The audio packet exchange between clients and CSs and between CSs is similar to what was discussed in Chapter 3. The solution proposed here closely follows the architecture using SIP messages. Clients may go out of the conference by sending usual BYE SIP messages. When there are no clients in a conference from a

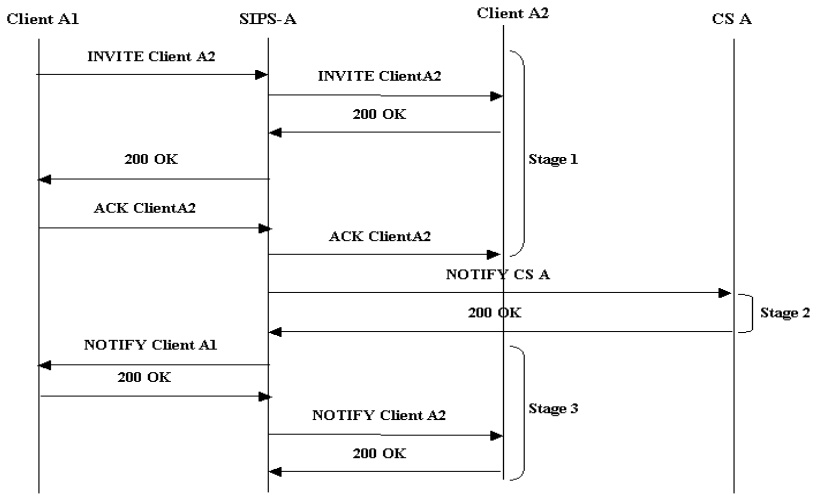


Figure 8.17: **Flow of SIP messages when Client A1 invites Client A2**

domain, SIPSs of that domain unsubscribe from all other SIPSs serving that conference. Similarly, a CS may unsubscribe from SIPS in their domains while closing down.

8.6 Allocation of Selectors

The architecture to support voice conferencing discussed in Section 3.5 did not consider the process of choosing selectors and the formation of a selector group with clients in its domain. Subsequently, Sections 3.6 and 3.7 dealt with scalability of the system and scalability with respect to the capacity constraint of a selector. It was implicitly assumed that the clients are naturally grouped with the nearest selector for conferencing support. It was also noted that by increasing the number of domains the net network traffic increases. How is one to reduce the number of selectors with least amount of traffic generated by clients connecting to them? (There is a limit to the number of streams that a selector can handle.)

Frequently asked questions are: (i) If a client is in a remote domain, say, a dial up link, how can that client be served? and (ii) Should a selector be opened in its domain or should it be assigned to a selector in another domain? In case of an isolated client, the nearest selector can serve it and multiple streams from selector to client can be mixed at that selector to reduce the bandwidth. (iii) Next, if there is more than one client but they are near and grouped, is it cost effective to assign a new selector to that group? These questions are addressed in this section.

There are some constraints to be considered to begin with. Selectors cannot be

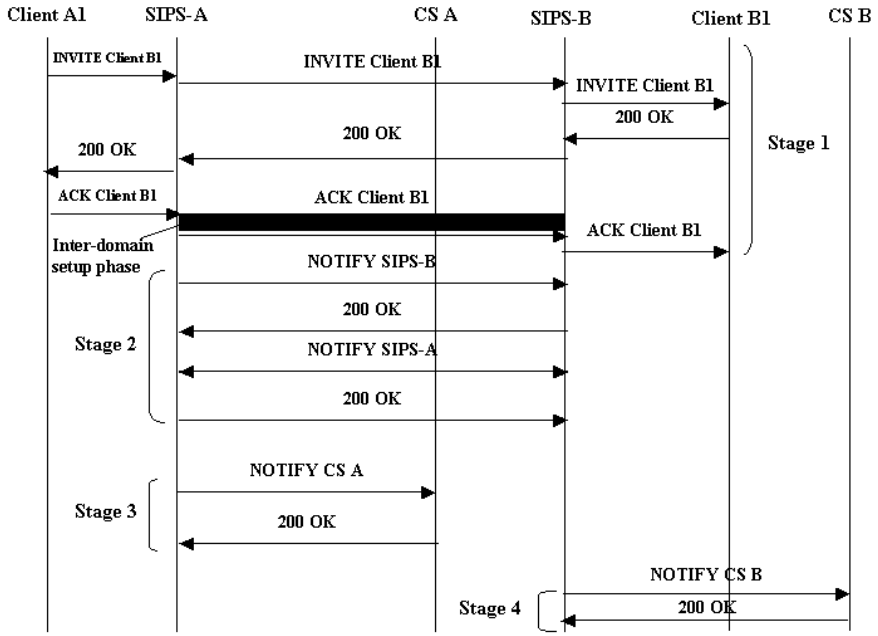


Figure 8.18: **Flow of SIP messages when Client B1 is invited by Client A1**

opened at all locations since they are specialised servers always connected to a control point such as a CP. Selector positions are at fixed locations in the Internet where a server is identified to run selector application. Selectors are opened whenever required at these pre-identified locations. Clients are to be assigned to only these selectors as and when required.

Before formulating the selector allocation (or location) problem, some discussion on the cost of service will be useful. Knowledge of the entire network and locations of clients and selectors is necessary for an overall gainful assignment of clients to selectors. Hence it is assumed here that some information of the parameters of the link connecting clients to selectors is available. Depending on the specific needs, the cost of the connection between a client and a selector can be defined in terms of number of hops, link bandwidth, and cost of the physical connection. In case the link is a leased line connecting to Internet, the cost of using that link can also influence calculation of the overall cost. Cost is not specifically defined in the following discussions but some notion of cost based on the above parameters is used. Similarly, hardware and location form the basis for computing the cost of opening a selector. These two costs – cost of connection between a client and a server, and cost of opening a selector – are used to find the allocation of clients to selectors. To discourage opening of a selector at a location, cost of opening that server can be made very high. Similarly, the cost of

connecting a selector to a remote client can be made very large. The costs taken here are conceptual.

A selector has to incur some computation for each client assigned to it depending on whether the client is a soft client or a hard client, the coding used by the client and whether mixing is required at the clients. These overheads, called demands from clients, can be taken as units of computation. Selectors have limited computation capacity.

CP can allocate selectors to clients before the start of a conference if the clients in that conference are known apriori. In case clients are successively invited to an ongoing conference, then CP has to reallocate selectors to clients. This can be done in two ways: (1) Only the new client is allocated to that selector which reduces the cost; (2) Periodically, all clients are allocated afresh to selectors available, thus reducing the overall cost. In VoIP conferencing, allocation of a selector to a client means that the client should send and receive packets to and from that selector. This involves changing the selector address stored at clients.

Allocation of clients to selectors with the above setting is popularly known as the Facility Location problem. Facility locator problems, in Operations Research deal with finding optimum number of shops to be opened so that total cost including opening cost of shops and transportation cost for customers is minimised. Here this is equivalent to selector maintaining cost and clients connecting costs. This class of problems is known to be NP-hard [230, 231].

8.6.1 NP-hard problems

The difficulty of a combinatorial optimisation problem is measured by the time required to solve the most difficult instances of the problem as a function of the size of the instance (input parameters). A problem that can be solved in time that is bounded by a polynomial in the size of the problem for even the most difficult instance is considered easy. A hard problem is one for which no such polynomial time bound algorithm exists, i.e., there is no algorithm to solve these problems that is qualitatively better than the explicit enumeration of all possible solutions. Garey and Johnson [232] popularised the term NP-hard for this class of hard problems. Almost all of the combinatorial optimisation abstractions motivated by real world problems, with a few exceptions, are NP-hard[4]. Problems of similar difficulty are then grouped into classes. NP-complete problems are one such class of problems. A problem in this class can be transformed

[4] "I can't find an efficient algorithm, but neither can all these famous people."

- Garey and Johnson [232]

to another problem in the same class in polynomial time [232]. Therefore finding a solution to one problem means finding solutions to all other problems in this class. NP-complete problems are well-defined compared to hard problems. Nonetheless, to find whether a new problem 'A' is NP-hard, it is sufficient to transform any known problem in NP to problem A in polynomial time [233]. Solving an NP-hard problem is difficult or at least as hard as any of the NP-complete problems. There are three well known ways of solving these problems [234].

The first is to construct algorithms that take exponential time for the most difficult instances, but that find the optimum fast enough for typical instances (smaller and easy versions of the hard problems). In practice, the most common examples of this approach are the back tracking and branchandbound [235] algorithms for solving integer programs. However, when an instance proves too difficult to solve quickly, the algorithm is stopped before proving optimality resulting in no solution at all or a solution of questionable quality.

Heuristics provide an alternative. Heuristics usually produce excellent solutions very quickly, provided the instances being considered happen to agree with the relevant intuition. The heuristic solution tries to exploit some known characteristics of a problem. Unfortunately, heuristics cannot guarantee a solution in all instances. It often depends on the context in which they are developed. Typical instances from a particular context often fail to cooperate all the time. Therefore many heuristic algorithms are run on the same problem instance and the best solution is taken. Each heuristic algorithm may be based on a different intuition or on variations in the methodology. Heuristic algorithms which provide a solution do not guarantee optimality. It is hoped that not all heuristic algorithms or its variants will perform badly on the instance of a problem.

The next alternative is approximation algorithms. These relax the optimality requirement, instead they provide a solution within some factor of polynomial time. They guarantee a reasonable running time and a good solution on even the most difficult instances. In practice they typically run faster than exponential time algorithms by sacrificing optimality. However, approximation algorithms are usually slower than heuristic approaches, but guarantee a good solution for instances that fall outside the domain of the available heuristics. Approximation algorithms are known to result almost always in solutions well within their guaranteed bound for the most difficult instances. Solutions from approximation algorithms are sometimes used as backups when integer programming or heuristic approaches fail, or as good starting solutions for heuristics that make incremental improvements. A heuristic algorithm is suggested

in this thesis for the selector allocation problem.

8.6.2 Selector allocation problem

Let $F_S = \{1, 2, \ldots, m\}$ be the set of m selectors situated at pre-identified locations. Let $D_C = \{1, 2, \ldots, n\}$ be the set of n clients whose locations are known. Let the cost of connecting a client $j \in D_C$ to a selector $i \in F_S$ be c_{ij}. The notion of cost follows the discussions in Section 8.6. Let d_j, $j \in D_C$, be the demand (in the form of computation requirement, in integer units) from Client j and let f_i, $i \in F_S$ be the cost of opening Selector i. Let l_i, $i \in F_S$, be the limit of Selector i in number of units of computation that it can afford.

Now the client allocation to selectors can be expressed as an integer program, with x_{ij} and y_i as $0-1$ integer variables. $y_i = 1$, $i \in F_S$, indicates that the Selector i is opened and $x_{ij} = 1$, $i \in F_S$, $j \in D_C$, implies that the Client j is assigned to Selector i. The minimisation problem ($\mathrm{P_{min}}$), is formulated as an integer program and is

$$\min : \quad \sum_{i=1}^{m} f_i y_i + \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{8.1}$$

subject to

$$\sum_{j=1}^{n} d_j x_{ij} \leq l_i, \quad \text{for each} \quad i \in F_S \tag{8.2}$$

$$x_{ij} \leq y_i, \quad \text{for each} \quad i \text{ and } j, \; i \in F_S; \; j \in D_C \tag{8.3}$$

$$\sum_{i=1}^{m} x_{ij} = 1, \quad \text{for each} \quad j \in D_C \tag{8.4}$$

$$x_{ij} \in \{0, 1\} \quad \text{for each} \quad i \in F_S, \; j \in D_C \tag{8.5}$$

$$y_i \in \{0, 1\} \quad \text{for each} \quad i \in F_S \tag{8.6}$$

Constraint 8.2 limits the number of clients connected to a selector depending on its limit. Constraint 8.3 ensures that if there is an assignment of a client to a selector then that selector must have been up and running. Constraint 8.4 ensures each Client j is assigned to a Selector i, $i \in F_S$ and $j \in D_C$. Constraints 8.5 and 8.6 are part of integer programming. Another constraint that a client is allocated to only one selector is captured in the above constraints and hence not mentioned explicitly. This constraint is called 'unsplittable' demands in operations research terminology. Note that the second term in ($\mathrm{P_{min}}$) is not multiplied by d_j, as is usually done if facilities are shops and c_{ij} are customer locations to cover cost of transportation.

This is a hard problem to solve since there is no polynomial solution known. There are actually two problems: (a) assigning clients to selectors with less cost; and (b)

opening least number of selectors so that there is saving in supporting the conference service. However, the assignment and number of selectors to be opened are to be found so that overall cost is minimised. This problem in its simplest form (where demands are splittable and with or without a limit on capacity of selectors (servers)) is solved using approximation algorithms. Some solutions consider unit demands. When demands are splittable, i.e., they can be met by more than one server, solutions using transhipment algorithms to assign the clients are suggested. These solutions and their variants are covered in [236, 237, 238, 239]. When the selectors (servers) have limits and clients are to be served by one selector (unsplittable demands) it is NP-hard to compute the minimum assignment even if the optimum selectors to be placed are given as the input [239]. Therefore the solution suggested here to this problem is heuristic based.

8.6.3 Heuristic algorithm

In this problem of allocating selectors, there are instances in which a solution may not exist at all. A case in point is when the total capacity of selectors is less than the total demand by clients. In these cases heuristic algorithms definitely fail. Heuristic algorithms try to find a near optimal solution, if there exists a solution. The difference between the solution found using a heuristic algorithm and an optimum one (found using IP formulation of the problem, say) is unbounded.

The algorithm suggested here is divided into two modes. The first one is the assignment mode. Here the clients are assigned to selectors in set $F_S^G \subseteq F_S$. Selectors only in F_S^G are considered; the algorithm considers selectors outside this set as nonexistent. There are three heuristic methods for this based on a greedy principle. In the second mode, the number of selectors used are minimised by closing some of them and clients previously assigned to selectors, which are closed now, are reallocated to other selectors. Algorithms of the first mode are given below.

Algorithm 8.1 ***Assignment of clients to given set of selectors (F_S^G)***
Arrange c_{ij} in list $L(k)$ in ascending order, where $k = \{1, 2, \ldots, mn\}$.
Each k corresponds to a pair (Selector i, Client j).
Consider assignment of clients to a selector as $A_C(j)$, a row vector that holds the selector identity i for each Client j.
$A_C(j)$ is zero vector to start with.
$k = 1$. *REPEAT*
{

Step 1: *Assign the Client j to Selector i that corresponds to k of $L(k)$ if*

- *For Selector i, $l_i \geq d_j$, and*
- *Client j is not already assigned;*

Step 2: *$l_i = l_i - d_j$, if Client j is assigned to Selector i. Mark Client j as assigned by updating $A_C(j)$.*

Step 3: *Stop if all clients are assigned; Return the total assignment cost (that is, sum of c_{ij} in the assignment) and $A_C(j)$.*

Step 4: *Return 'Infeasible' if all clients are not assigned when $k = mn$.*

Step 5: $k = k + 1$

}

Algorithm 8.1 is the popular greedy solution wherein the client nearest to a selector is assigned first. This heuristic may give an optimal solution in some instances [240]. The complexity of this algorithm is $O(m^2n^2)$. The next algorithm is an intelligent assignment solution.

Algorithm 8.2 ***Assignment of clients to given set of selectors (F_S^G)***
Consider matrix $[c_{ij}]$ wherein each row, i, represents a selector and each column j represents a client.
Consider assignment of clients to a selector as $A_C(j)$, a row vector that holds the selector identity i for each Client j.
$A_C(j)$ is zero vector to start with.

Step 1: *Find $b(j) = \min_i\{c_{ij}\}$ for each $j \in D_C$.*
That is, finding the Selector $i \in F_S^G$ that results in minimum cost of assignment to Client j. $b(j)$'s have the minimum cost of assigning Client j to a selector.

Step 2: *Update $A_C(j)$ by assigning Client j to Selector i that corresponds to the minimum cost found above. Note: If two positions in $[c_{ij}]$ have minimum value, the first one is chosen.*

Step 3: *Find set of selectors, $F_S^{(OL)}$, that are overloaded taking into account $A_C(j)$ and l_i. If there are no overloaded selectors, goto Step 7.*

Step 4: *Find difference matrix $[a_{ij}]$*
where $a_{ij} = c_{ij} - b(j)$, $i \in F_S^G$, and $j \in D_C$.
Mark 'X' in $[a_{ij}]$ if Client j is allocated to Selector i in Step 2.

Step 5: *For each Selector* $r \in F_S^{(OL)}$,
for clients $s \in D_C^{(r)}$, *i.e., clients assigned to the Selector r,*
find new selector-client pair (p,q) that has the position of
$\min_{(i,s)}\{a_{is}\}$, *(i.e.,* min *over i and s),*
where $s \in D_C^{(r)}$, $i \in F_S^G$ *and* $i \notin F_S^{(OL)}$
and Selector p is able to serve this new client
without overloading itself
(a_{ij}'s marked with 'X' are not to be considered)

Step 6: *If a new selector is not found in Step 5, Return 'Infeasible'.*
Else, assign Client q to Selector p.
Subtract the load d_q *from Selector r.*
Update $A_C(j)$ *and* $F_S^{(OL)}$ *taking into account this change.*
Mark a_{rq}, *the previous position of Client q, with 'X'.*
Repeat Step 5 till $F_S^{(OL)} = \{\}$.

Step 7: *Return* $A_C(j)$ *and cost of assignment.*

Heuristic Algorithm 8.2 has some intelligence in it. Minimum in each column of $[c_{ij}]$ is found first. This minimum is subtracted from other values in $[c_{ij}]$. This makes the difference matrix $[a_{ij}]$. First, clients are allocated to selectors considering only the minimum cost. If no server is overloaded, the assignment is found. If not, some clients from the overloaded selectors are moved to selectors that are not overloaded. Reassignment is done with small, hopefully least, increase in cost. Also, the new selector to which a client is going to be reassigned will not become overloaded after reassignment. This heuristic algorithm has worst case complexity of $O(m^2n)$.

Algorithm 8.3 ***Assignment of clients to given set of selectors*** (F_S^G)
Note: All the steps here are similar to those in Algorithm 8.2 except Step 5. Step 5 here is given below.

Step 5: *For each Selector* $r \in F_S^{(OL)}$,
for clients $s \in D_C^{(r)}$ *i.e., clients assigned to the Selector r,*
find new selector-client pair (p,q) that has the position of
$\min_{(i,s)}\{a_{is}\}$ *(i.e.,* min *over i and s),*
where $s \in D_C^{(r)}$ *and* $i \in F_S^G$
(a_{ij}'s marked with 'X' are not to be considered)

Algorithm 8.3 is a minor variant of Algorithm 8.2. Here a selector will be assigned even if it is overloaded already or becomes overloaded after assignment. Each selector

sends some client to other selectors to reduce the load without bothering about the load on other selectors. In the process, this selector might get overloaded again since other selectors might send some clients back to it. A selector which is overloaded always selects a new selector for one of the clients in its domain such that transferring that client results in less extra cost compared to that of other clients. The difference between this algorithm and the previous algorithm is only in Step 5 shown here. This heuristic algorithm has worst case complexity of $O(m^2n^2)$.

All the above algorithms in the allocation mode can be used individually or each one may be applied on an instance and the best possible solution used. Depending on the problem instance one may perform better than the other. These algorithms are used in the main heuristic algorithm to solve the problem $\mathrm{P_{min}}$ in the second mode.

Algorithm 8.4 ***A heuristic solution to the problem (P_{min})***

Let $F_S^{(O)}(i)$ be a vector that represents whether a selector is open. It is found using client assignment vector $A_C(j)$.

$$F_S^{(O)}(i) = \begin{cases} 1 & \textit{if Selector } i \textit{ is serving any client} \\ 0 & \textit{Otherwise} \end{cases}$$

Total cost $SOLN$ of any assignment is found using

$$SOLN = \sum_i F_S^{(O)}(i) f_i + \sum_{\forall i,j} I_C(j) c_{ij} \tag{8.7}$$

$$\textit{where} \quad I_C(j) = \begin{cases} 1 & \textit{if Client } j \textit{ is served by Selector } i \\ 0 & \textit{Otherwise} \end{cases}$$

Step 1: *Initialise $A_C(j)$and $F_S^{(O)}(i)$ $\forall i, j$ to zero, $F_S^G = F_S$ and a set $Z = \{\}$.*

Step 2: *Find $A_C(j)$, $F_S^{(O)}(i)$ corresponding to a minimum $SOLN$, after using three Algorithms 8.1, 8.2 and 8.3 of first mode.*

Step 3: *Find T_i, $i \in F_S$, as defined below and arrange in descending order.*

$$T_i = \frac{f_i}{l_i} - \sum_i \sum_j I_C(j) c_{ij}$$

Step 4: *For $k = 0$*

Do {

Temporarily close Selector u

that corresponds to the highest T_i, $u \notin Z$

(a) Find the best $iSOLN$ using Equation 8.7

Table 8.1: **Heuristic Algorithm 8.4 compared with LP solution**

Number of Selectors	Number of Clients	Range of l_i	LP Solution	Heuristic Algorithm 8.4 with Algorithm(s)			Ratio of Heuristic to LP
				8.1	8.1 & 8.2	8.1, 8.2 & 8.3	
10	50	1-50	9707	10309	10309	10309	1.062
10	100	1-50	13000.4	13207	13037	13037	1.003
10	150	1-150	20439.6	22620	21155	21015	1.028
10	500	1-500	53541.5	57047	53876	53852	1.005
10	1000	1-1000	99698	102049	99951	99951	1.003
10	1000	1-800	100376	112422	101813	101734	1.014
10	1000	1-1000	105684	127217	111214	111420	1.052
10	1200	1-1000	174134	179141	174443	174447	1.002
12	1000	1-1000	93761.8	102985	94732	94794	1.011

(Here iSOLN means intermediate solution)
out of three returned assignments $A_C(j)$'s of three first mode algorithms with $\{F_S^G - \{u\}\}$ as selectors open for assignment.
If ($iSOLN < SOLN$)
then permanently close Selector u and $F_S^G = F_S^G - \{u\}$;
$SOLN = iSOLN$;
Recalculate T_i as above with Selector u closed and arrange in descending order;
Else, $Z = Z \cup \{u\}$;
(b) $k = k + 1$;
}*Repeat till $k = m$;*

Step 5: *$A_C(j)$ of $SOLN$ corresponds to the best assignment found by this algorithm. The total cost is $SOLN$.*

This algorithm takes into account all the first mode algorithms and the best assignment of the assignments returned by them is considered at each stage (see Step 2 and Step 4). There can be many variants of this algorithm where one, two, three or some combinations of the first mode algorithms are used. The complexity of this algorithm is m times the complexity of the first mode algorithm. For example with all three first mode algorithms, complexity is $O(m^3n^2)$ since Algorithm 8.3 has the highest complexity.

8.6.4 Results

Linear Programming (LP) formulation of (P_{min}) relaxes the constraints that x_{ij} and y_i be integers. Therefore the solution found by LP is the best that one can find. Integer Linear Programming (ILP) is difficult to solve and if there is a solution to (P_{min}) it would cost greater than LP solution. Also, Linear Programming formulations are complicated [239], though solvable in polynomial time. However, to compare the proposed algorithm, LP solutions on some test cases are found using LP_SOLVE programs freely available in the Internet. It is interesting to see how Algorithm 8.4 fares for some test samples. In the test samples cost matrix c_{ij} is generated randomly with entries between 1 and 1000. Computation demands d_j were randomly chosen between 1 and 5. Selector opening cost is chosen to be randomly between 1 and 1000. Capacities of selectors are changed across test samples and the range is given in Table 8.1. The table also gives the performance comparison of the algorithm with that of the LP solution, which is the lower bound.

The optimal cost of the solution for the problem (P_{min}) actually lies between that of LP solution and of the heuristic Algorithm 8.4. The last column of the Table 8.1 shows the ratio of performance between LP and the suggested heuristic algorithm. Therefore, the difference between heuristic solution and the "optimum" value (found by ILP) would be even less. Algorithm 8.4 is implemented in three different forms differing in the method of assignments in first mode. They are: (a) Algorithm 8.1 alone; (b) Better of Algorithm 8.1 and 8.2; and (c) Best of Algorithms 8.1, 8.2 and 8.3. For calculating the cost ratio of the heuristic solution to that of the LP solution, the best result of Algorithm 8.4 in three forms is used.

It can be seen from the table that even for large instances of the problem the error is reasonable. It can also be seen that with two heuristics used for assignment, Algorithm 8.4 achieves results very close to the optimum solution compared to using Algorithm 8.1 alone for assignment mode. The order in which the first mode algorithms are used also matters. The heuristic may not give solutions in a few cases though a solution exists. In case a heuristic algorithm fails, then assignment has to be based on some other method. Nonetheless, heuristic algorithms serve as a good starting point to group clients.

8.7 Limitations

This chapter dealt with some of the implementation issues. The protocol suggested here is actually developed for the proprietary application developed in-house. Due

to this, the applications cannot inter-operate with other standard VoIP applications available in Internet.

Algorithm for managing streams with synchronisation presented in this chapter works well with WinNT systems and Linux systems since the "timer" resolution is better compared to Win95/98. It fails when the delay jitter is high because of the inherent buffering process. However, the management of the streams is possible with the same algorithm foregoing the requirement of stringent synchronisation. It is observed that inside a sub-net it works well. Design of components such as CP and selector are useful for a high level design of the system. The design presented here focuses on the architecture proposed in Chapter 3. Results of the use of this application with many clients, ranging up to hundreds, are not available yet because of limited resources.

The proposal for using SIP in this architecture as well as for inter-operability with other SIP clients and/or proxies is not tested yet. Due to SUBSCRIBE/NOTIFY messages, the total traffic in the network is high because of control messages. There is a need to reduce these messages. Though the selector allocation problem discussed here allows system administrators to use a tool for assigning clients, it is still in infancy. This is due to the abstract nature of costs defined in this thesis. Heuristic algorithms need to be fine tuned for near optimal results. However, there is no guarantee that this heuristic algorithm works for all instances of the allocation problems. A theoretical bound on these heuristic algorithms is not yet available.

8.8 Conclusions

This chapter gives the big picture of all the discussions of this thesis. The ideas are tested after implementing them on Windows platform OS. The results are highly encouraging with CP, selectors, client applications, PSTN Gateway and VMS. PSTN gateway and VMS were also implemented and tested (not discussed here since it is outside the purview of this thesis). A VoIP test-bed is now available for experimentation with all services including PBX support, VMS and PSTN connectivity. The test-bed implementation holds testimony[5] to the concepts expressed in the initial chapters though on a small scale.

The SIP proposal is promising for inter-operability and wide acceptance of this

[5]It's an experience like no other experience I can describe, the best thing that can happen to a scientist, realising that something that has happened in his or her mind exactly corresponds to something that happens in nature. One is surprised that a construct of one's own mind can actually be realised in the honest-to-goodness world out there. A great shock, and a great, great joy.

- Leo Kadanoff

architecture. The selector allocation algorithm is a novel and simple short hand tool for managing large conferences of tomorrow. The heuristic algorithm presented here is a useful tool which works in polynomial time to support realtime allocations.

Chapter 9

Summary and Conclusions

> In five minutes you will say that it is all so absurdly simple.
>
> - Sherlock Holmes, The adventure of the Dancing men
> - Sir Arthur Conan Doyle

9.1 Recapitulation

The first chapter records the trail of an effort aimed at forming a perspective of the vast, emerging and promising field of VoIP as a medium for multi-party interaction with emphasis on audio conferencing. As the studies evolved, it was recognised that there were many unsolved issues in the state of the art as reported in literature, a few of which could be refined, some adapted more gainfully, some others circumvented and yet others seen from different angles. The first chapter documents a critical exploratory journey that eventually served to ensure that the problem chosen to be investigated here indeed had scope for research and development. Consequently, VoIP was critically analysed vis-à-vis telephony services which are mainly supported by telecom companies using dedicated circuit switched networks.

It is felt that any research has one or more of the following main challenges. First, a system theoretic challenge that leads to better understanding of a field, exposes the hidden ignorance, serves to intellectually satisfy the researcher and hence provides value addition to the body or literature. Secondly, it could provide a technical backbone throwing potentialities open to be leveraged into technologies in the immediate or distant future. Finally, the actual realisation of dreams into realities – providing new technology resulting in enhanced 'quality of life' in a social environment.

In today's world, with economics and economies playing vital roles in the international scenario, it is particularly significant to pursue technology-dictated or technology-moderated research. Information Technology supported by computing machines on the one hand and communication on the other have ushered in the era of

Internet. Therefore, the future of VoIP applications, taking into account the market projections were discussed. There is a strong feeling that impelled by the growth of Internet, VoIP will become widely deployed in the near future, and will stay for long. Later, the main technological challenge, addressed in this thesis – of providing many-to-many conferencing support was discussed briefly to find its worth in its applicability to business and collaborative works.

In Chapter 2, the challenge of providing interactive audio conference service on the existing Internet was discussed by identifying a few main issues that are necessary to characterise this realtime service on IP network. Many existing techniques, standards and implementations were discussed to show the different standpoints taken in this thesis in providing conference service. It was recognised that satisfying all the requirements raised by those listed issues is a difficult proposition. After all, the requirements are conflicting; else, there would have been no charm in the effort!

It was observed additionally that there are two main schools of thought with respect to providing interactivity among participants. The first one assumes that participants take turns under global supervision and control, so that only one speak at an instant of time. Though usage of resources is more efficient in this case, it results in a gagging feel for participants who want to interrupt the present speaker. It hinders spontaneity. The second school suggests that interactivity and spontaneity be extended 'unbiasedly' to everyone in a conference to the extent possible. This demands more resources, mainly bandwidth. It is not desirable or even necessary to provide unbridled freedom to the participants to interact. Reported implementations were discussed. An important objection raised in this thesis targets taking one of the two extremes, namely, with tight control and a free-for-all service. The answer lies in recognising that the former is insensitive to the participants' feel for interactivity whereas the latter is blind to the economics of providing network connectivity and associated firmware. In reality, a hierarchical mixing scheme where all streams are mixed results in poor quality due to loss of spatialism through mixing. Though highly scalable and interactive, it increases delay due to intermediate mixers.

The solution to the problem of providing interactive conference needs to take human factors, socio-psychological behaviour of humans into account apart from implementation aspects, scalability and effective usage of the resources. Thus the problem is viewed differently here. That is, what is possible best that can be offered to a participant so that a reasonable balance is struck between interactivity and efficient usage of resources. The outlook of the solution is to give the participant what is necessary. Further, technology should be driven by a view to provide more and more intelligent

solutions, making the service user-friendly.

A foundation for realising an ambitious plan was laid in Chapter 3 in the form of an architecture to cater to the requirement of a scalable and interactive conference that takes into account most, if not all, issues that were raised. The architecture is such that it supports a large number of clients without proportionally scaled up demand on resources. It was supported on a conjecture and a dynamic quantifier proposed later. It was shown that this architecture scales well compared to completely distributed and centralized solutions even as it takes into account all relevant practical issues.

The problem of remote conferencing on network, by nature, involves the feel of the participants. This is the backdrop in which Chapter 4 is set. As the transactional medium is exclusively voice here, it revolves around an estimated net-audio-presence of other participants. Therefore, it is imperative that considerable weightage be given to the socio-psychological aspects of behaviour of humans in conversation.

The question was: 'How many simultaneous speakers need be allowed, and how, in a blind conversation?' Though the debate is interdisciplinary and hence novel in the context of voice-only conferencing on VoIP, the simplicity of the result of Chapter 4 is one of its highlights. Fixing the number of floors is pivoted on findings from conversational analysis, a branch of ethnomethodology. It is assumed that users are mature enough to resolve the conflicts themselves and hence do not need the clutches of tight floor control schemes. As a rearguard safety measure in the rare event of the participants 'misbehaving', conflict-resolving repair mechanisms are invoked.

The result is labelled *conjecture* not because of the apprehension that its content is actually any less rigorous, but due to the fact that its basis extends to ethnomethodology, sometimes rather unfairly classified among the so-called soft sciences[1].

Chapter 5 proposes a new quantifier, Loudness Number, for each speaker so that a set of speakers is selected for playout dynamically. The design of loudness number is such that it allows a current speaker to continue if active. It avoids frequent floor changes to bring in some discipline since no controller is used to allot floors to the users. It is observed that the conference maintained by using this parameter is more user-friendly. The number of participants in conferences considered in pilot runs were insufficient[2] to make a definitive statement on the use of loudness number. However, loudness number along with the conjecture on fixing the number of floors is fresh in the design of computer-human interface applications. There is no pretension of providing a

[1]Indeed, quantifiability is not the exclusive forte of reasoning! Inductive logic is a glorious counter-example.

[2]The man who insists on seeing with perfect clearness before he decides, never decides.

- Amiel's Journal

solution using loudness number which matches exactly the feel of a real-life conference in the existing form alone.

The next natural step is to refine the solutions to render them less demanding on resources. Bandwidth reduction algorithms proposed in Chapter 6 is a significant step in this direction. It uses the fact that, more often than not, in serious conferences, users resolve conflicts – using N_{Max} floors – as to who should speak. Therefore, for a significant fraction of conference time only one would be addressing. This fact, in tandem with loudness number, is harnessed to reduce the traffic further. Two heuristic algorithms are proposed in this connection.

An additional feature is incorporated by noting that speech has silent spells. This is leveraged to reduce traffic generated by individual participant. The VAD algorithms proposed demonstrate a considerable saving in bandwidth with reasonably good quality speech. Computational reduction comes as a significant bonus.

It was recalled in Chapter 7 that the proposed solution in this thesis assumes availability of some reasonable bandwidth for the application deployment. The solutions proposed are only at the application layer. This has an advantage since it decouples the application layer from the lower layers. If the application manages reasonably without getting any QoS guarantee from network then it would surely fare better with QoS support. In this sense tracking of some network effects such as packet loss and delay would be useful. Some experiments and analysis reported here deals with the effect of packetisation interval. It was seen that if packetisation interval is more, the effect of delay jitter is less. Simulation model may be used to find the applicability of the techniques here for future improvements.

Chapter 8 considers protocols and related issues, since the problem addressed here is service oriented in its true sense. Design of the application discussed, along with screen captures (Appendix A), gives a feel of applications implemented on the test-bed. This chapter deals with proprietary protocol for prototyping of the service, working of the selector for packet selection and packet stream management. Discussion on the design of applications (only servers in this thesis) and server database while handling a conference showed the complexity involved in implementation. A proposal for porting this application on SIP standard was discussed with an example of a conference setup.

A heuristic algorithm presented here tackles the well known NP-hard facility locator problem for allocating clients to selectors. This algorithm was shown to be performing well as the solution is very close to the solution of LP formulation of the problem.

In essence, this thesis reports an alternative, promising and novel solution to the problem of supporting voice conference on VoIP. In the light of these discussions this

approach is not merely a new methodology related to audio conferencing on VoIP; more fundamentally, in approaching multimedia conferencing on VoIP it presents a shift in paradigm.

9.2 Self-criticism

Efforts so far have been directed mainly at advocating[3] the material presented in this thesis. The stage is now set to step back and have a critic's[4] view of the thesis to the best possible extent. After all any intellectual work must be considered infirm if it is not dissociated from personal considerations. Introspection is now attempted on an issue-by-issue basis.

On the problem addressed:

In providing solutions to an engineering problem a possible question that arises is, 'Has the problem been blown out of proportion to make it seem worthy of non-trivial investigation?'

It is a positive outlook to solve what *needs* to be solved rather than what *can* be solved. The problem identified in this thesis has a technological flavour, is product-oriented and therefore addresses implementation. The work here addresses audio conferencing support over Internet. Many related problems have been widely studied [27, 28, 40, 61, 45, 47, 48] taking different views of the problem. As a matter of fact, the issues addressed in literature are based on assumptions not quite realistic. In the light of their investigations, the approach to problem formulation here is different from that in others, and hence is significant.

Work undertaken here deals with a problem that is close to reality. It is necessary to propose solutions to all the problems connected with a major class of requirements. Moreover, in any investigation – however small or insignificant the solution may be – new approaches taken towards solving the problem make the field of study interesting.

On Loudness Number:

Loudness number is developed heuristically. Setting values for parameters, controlling, and testing are non-standard and qualitative; the measures of performance include subjective considerations. Would it not have been desirable to have a quantitative machine evaluation?

[3] It is part of a good general to talk of success, not failure. - Sophocles, *Sayings*

[4] Don't speak in front of my Back! - Navjoth Singh Sidhu, Cricket Commentator

The field of study here mainly deals with Computer Human Interface (CHI). At the end of the day in this field it is not expected of the computer to certify a user's need; that would be topsy-turvy! It is the user who has to be satisfied with the package. Therefore an exclusively quantitative assessment is not only insufficient but also undesirable. Subjective aspects such as perception of the physical world presented by programmed computers are principle components. Thus the logic, design and implementations revolve around expectations and experiences rather than rigid non-subjective quantifiers. The tail should not be allowed to wag the dog! One may propose standardising quantitative measures by adopting double blind procedures in documenting human reactions. That indeed is the way ahead of here. Though no participant was consciously, or by intent, prejudiced in favour of or against any view, it must be conceded that the recordings here were not *clinically* double blind. The criticism is meet in this aspect.

On the method of fixing the number of floors:

Two pertinent criticisms on this result: (a) use of a conjecture; (b) a simple and common sense argument would have been sufficient.

It is in fact sufficient to consider qualitative analysis for fixing number of floors. Field of CHI deals mainly with the analysis carried out on many of the real world situations. A design or a result, which is directly connected with CHI, must involve results from its own field. Without the insights from the socio-psychological literature the results are incomplete. Though the end result may seem naive, it must be conceded that it has not been arrived at through a sequence of trivial considerations[5]. Evidently, no reference to this aspect is available in literature. Even H.323 does not explicitly fix a number for this. In this sense, it is better to argue, discuss and resolve it before moving ahead rather than assuming it without a debate. To be taken seriously by the scientific community, any solution that might have been arrived at using a common sense approach needs to be argued first. It requires explanations for such a solution based on the related investigations.

Overall Assessment:

There seems to be a lack of focus as this thesis dabbles with too many issues.

To have an acceptable service or a useful product, several facets will have to be

[5]The simplicity of a result is unquestionably a complimentary aspect; at any rate, it deserves no apology on that count. Do not trivialise a complex issue, as in the method adopted here. Do not complicate a simple issue, as in the result arrived at here.

looked into. Smoothly polishing some areas and leaving rough patches here and there does not make an attractive service module. Every non-negligible specification of a product, however simple it may be, needs to be met. It is desirable to find better solutions to more and more challenges than to propose excellent solutions to but a few. Some holes plugged very securely but just a couple of them left untended will eventually sink a boat! What has been consciously borne in mind here is to address several, if not all, challenges to the extent possible within the constraints of a thesis. It must be submitted with all humility that this modest effort is a sure step, not the last, in this direction.

9.3 Vistas for Future

Looking back at the solution given here are for an interactive VoIP conference, the grand goal is to ensure more effective large scale deployment. Given here is a conglomerate of individual problems that need further attention.

Architecture

The architecture for conferencing considers a simple model of the selectors and a single CP. A truly distributed architecture would span over many subnets and eventually many CPs need to be in place. Though there is a proposal at the end to work with many SIPS servers (similar to CP) using SIP, there is a need to understand the flow of messages across many subnets.

Further, when selectors serve many conferences, need to balance the load becomes an important issue. Blocking probability of conference requests will be an important issue that has to be tackled.

Refined Model for Conversational Analysis

Conversational analysis used to support the argument for fixing the number of floors considers the literature by Sacks et al. [123]. It would be interesting to have a fresh analysis of conversations of a blind conference. It is hoped that this study may provide some more insights.

Pilot runs of large scale conferences

Subjective quality of a conference reported here is for conferences having nearly ten participants. As it is expected that this system supports layer sizes, it needs to be substantiated by pilot runs of conferences with higher number of participants.

Refining Loudness Number

There were some discussions on the Loudness Number computation in Section 5.9. Many ideas presented there can be used to refine loudness number. For example, Loudness Number can incorporate the tone-fall at the end of utterances. It will be interesting to see after refinements whether loudness number adapts better to the dynamics of conversations of a conference.

Modelling playout buffer and Network

Models of playout buffer and network took a simplistic approach here. It is worthwhile to deliberate on the analytical solutions for these models that takes into account delayed playout. It is non-trivial to solve the finite queue size systems which takes into account variable packet sizes while determining the occupancy of the buffer.

Facility Locator Problem

Allocation of selectors to clients is a NP-hard problem. It is very challenging to find some approximate algorithms which give a bound on the solution. Finding a theoretical bound on the solution of the proposed heuristic algorithm may be a good starting point.

On dealing with packet loss and delay

It is necessary to experiment to circumvent packet loss and delay using various methods. As the approach taken in this thesis is to see the network effects and playout algorithms in toto, further studies – of effects of loss concealment, of coding taken with its effects, of packet interleaving and packet segmentation on the speech quality would be useful.

SIP Implementation

SIP implementation of the above proposal is a necessary future work to make this work inter-operable with other SIP clients and SIP Gateways.

Using Avatars and transmitting non-verbal cues

Perceived quality can be enhanced by using low bandwidth graphics called Avatars [241]. Integrating these into the present application and managing with selectors is an important step towards increasing the feel-good factor.

Singing off on a personal note

Philosophers have long since been really fascinated by dreams. Indian philosophy has always taken a great interest in the phenomena of dreaming and of dreamless sleep. There is something fascinating about the dream experience when it is analysed as Vedanta does. The dreamer may experience all sorts of things which he may not in the waking state: the blind may see, the one-legged can run, the beggar can be a prince. The dream experience is *real* during the dream! This then is the state of existence where, inter alia, several desires may be realised ...

The work reported here looks like one such dream. I found it *real* throughout. Now? As I rub my eyes at the transition from slumber, the realisation that it was all but a figment of my glorious imagination is beginning to dawn. Gentle reader! Do you intend to appreciate what I have blabbered here? Go take a nap! ZZZZZZZZZZ!

As I draw the curtains down relieving you of this bulk, I cannot resist the temptation to quote the last sentence of the thesis of my friend. Hmmmmm! Have a good day!

"I fell asleep reading a dull book, and I dreamt that I was reading on, so I woke up from sheer boredom." – Anonymous

Signing off on a personal note

Philosophers have long since been really fascinated by dreams. Indian philosophy has always taken a great interest in the phenomena of dreaming and of dreamless sleep. There is something fascinating about the dream experience when it is analysed as Vedanta does. The dreamer may experience all sorts of things which he may not in the waking state: the blind may see, the one-legged can run, the beggar can be a prince. The dream experience is *real* during the dream! This then is the state of existence where, inter alia, several desires may be realised ...

The work reported here looks like one such dream. I found it *real* throughout. Now? As I rub my eyes at the transition from slumber, the realisation that it was all but a figment of my glorious imagination is beginning to dawn. Gentle reader! Do you intend to appreciate what I have blabbered here? Go take a nap! ZZZZZZZZZZ!

As I draw the curtains down relieving you of this bulk, I cannot resist the temptation to quote the last sentence of the thesis of my friend. Hmmmmm! Have a good day!

"I fell asleep reading a dull book, and I dreamt that I was reading on, so I woke up from sheer boredom." – Anonymous

Appendix A

Screen Captures of Applications

A.1 Call Processor User Interface

Graphical User Interface (GUI) for the CP is shown in Figure A.1. N_{Max} value is set for all conferences and is communicated to selectors during start-up to select N_{Max} streams. "One minute ticks" shows the duration of running time of CP in minutes.

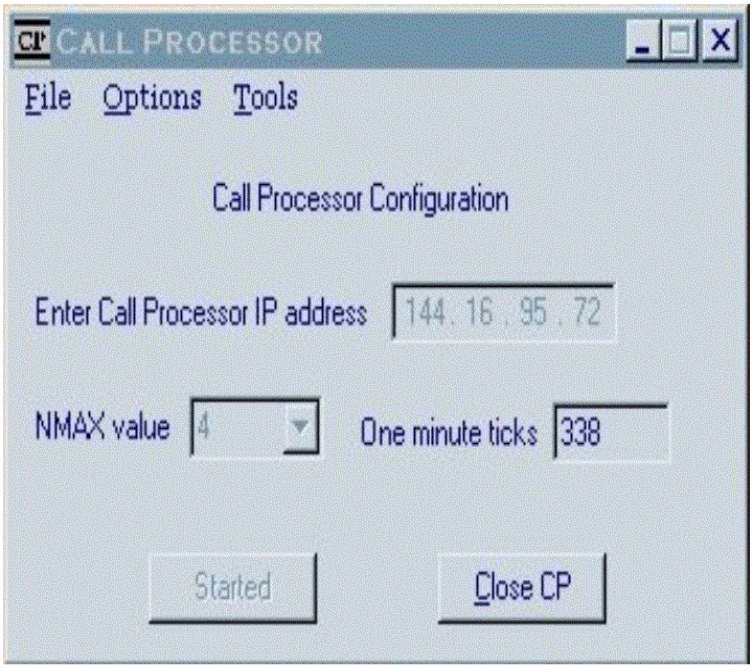


Figure A.1: **GUI of a CP**

A.2 Selector User Interface

User Interface of a selector is shown in Figure A.2. It shows the statistics for every conference it serves. The figure A.1 shows the statistics for the Conference #4.

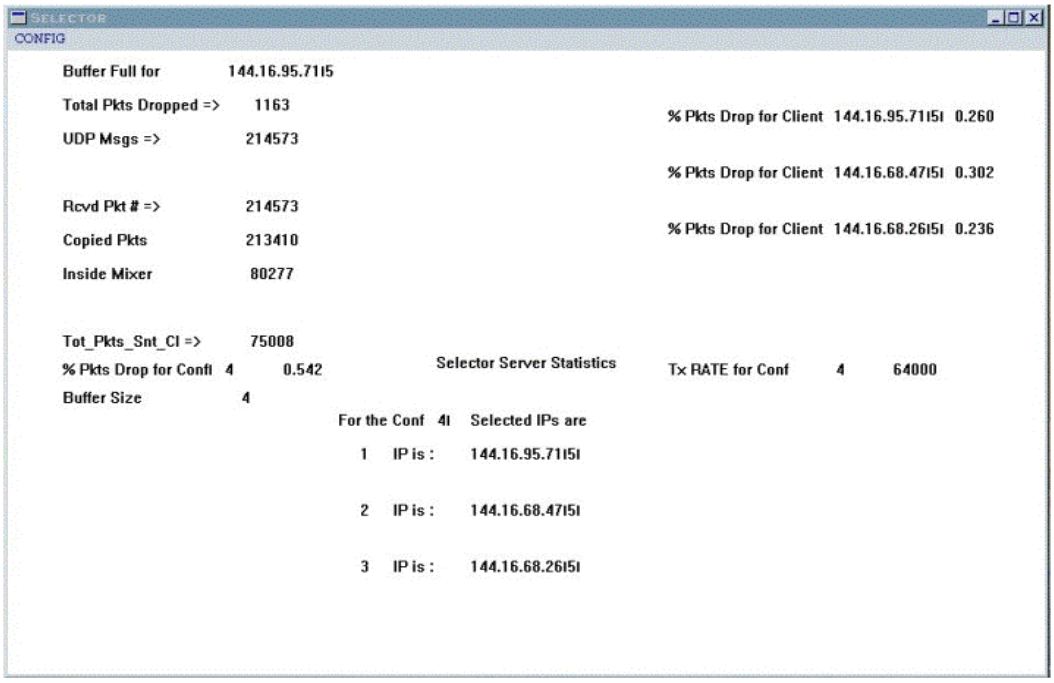


Figure A.2: **GUI of a selector**

A.3 Client User Interface

Figure A.3 shows a Client User Interface during client application start-up. User has to enter the email address and the DNS Name or IP Address of Call Processor and the DNS or IP Address of a Selector (in later versions CP allocates a selector).

A.4 Loudness Number Parameters

Figure A.4 shows the GUI for parameters for Loudness Number settings. There are three windows sizes; Recent Past, Distant Past and Activity Horizon. The weight Alpha 1 corresponds to the window Recent Past and Alpha 2 corresponds to Distant Past. The main window shows a conference which is live and a one-to-one call.

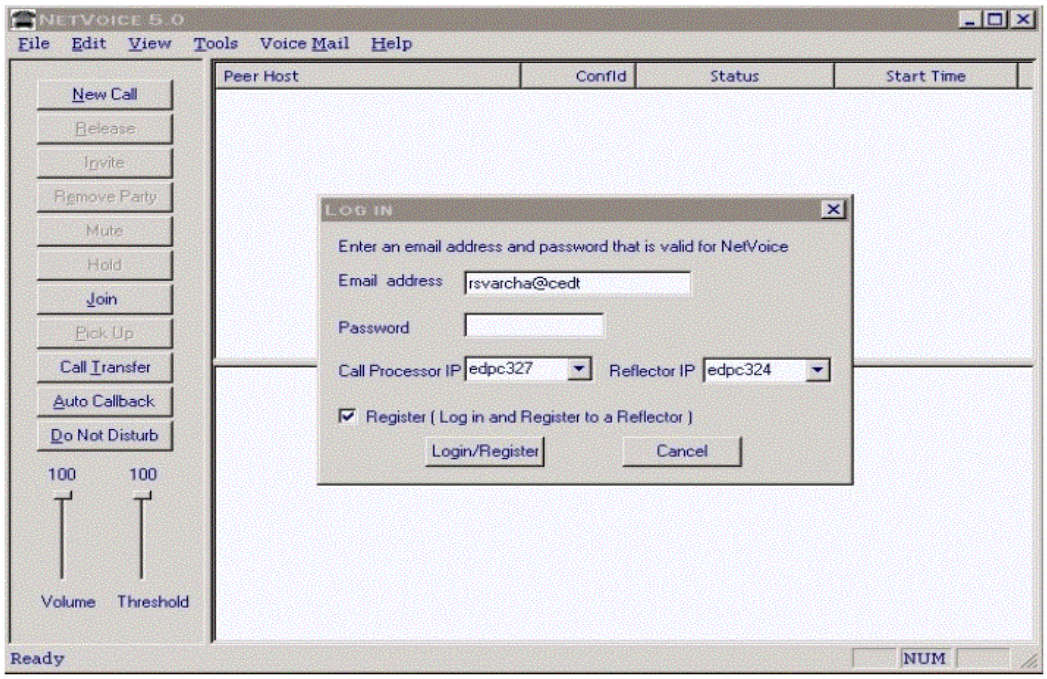


Figure A.3: **Client GUI during start-up**

A.5 Status Report and Recording Options

Figure A.5 shows the main client window with two other sub windows. The right side sub window shows the status report of a conference displaying number of packets sent, received, played, dropped etc. The left side window shows the recording options that can set by the user. At present the user has selected $\mu-$Law companding.

A.6 Customised Mixing

This GUI shows how to set the weights for individual sspeech streams from N_{Max} users in a conference selecetd by selectors. The email IDs of N_{Max} users selected appear corresponding to a slider bar. One can move the slider and set the weights for that participant.

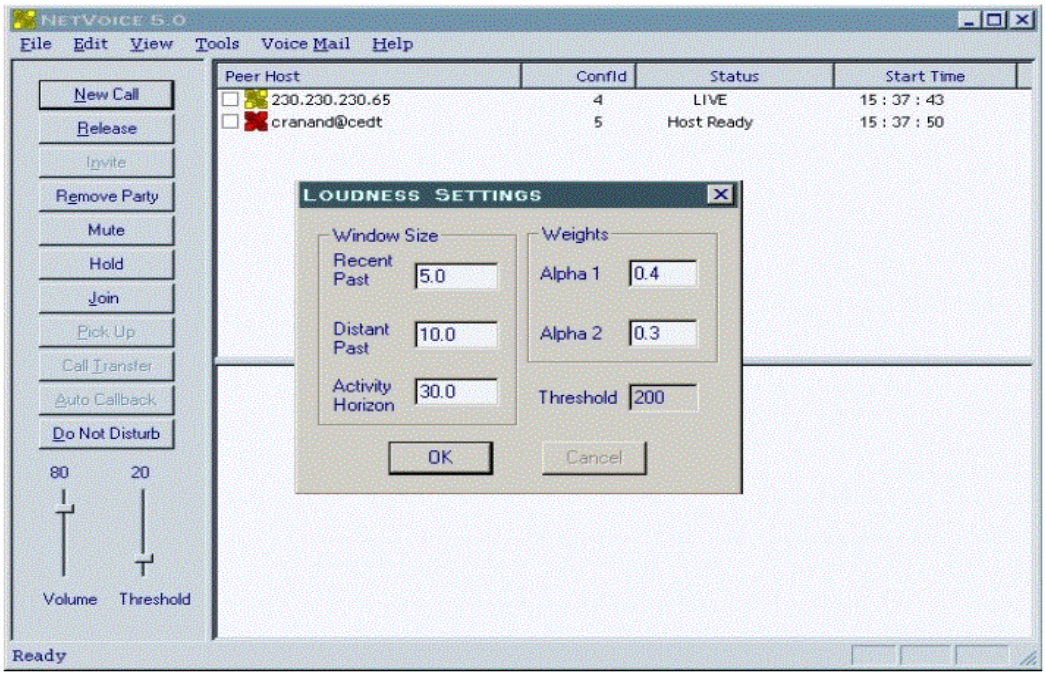


Figure A.4: **Loudness number settings at a client**

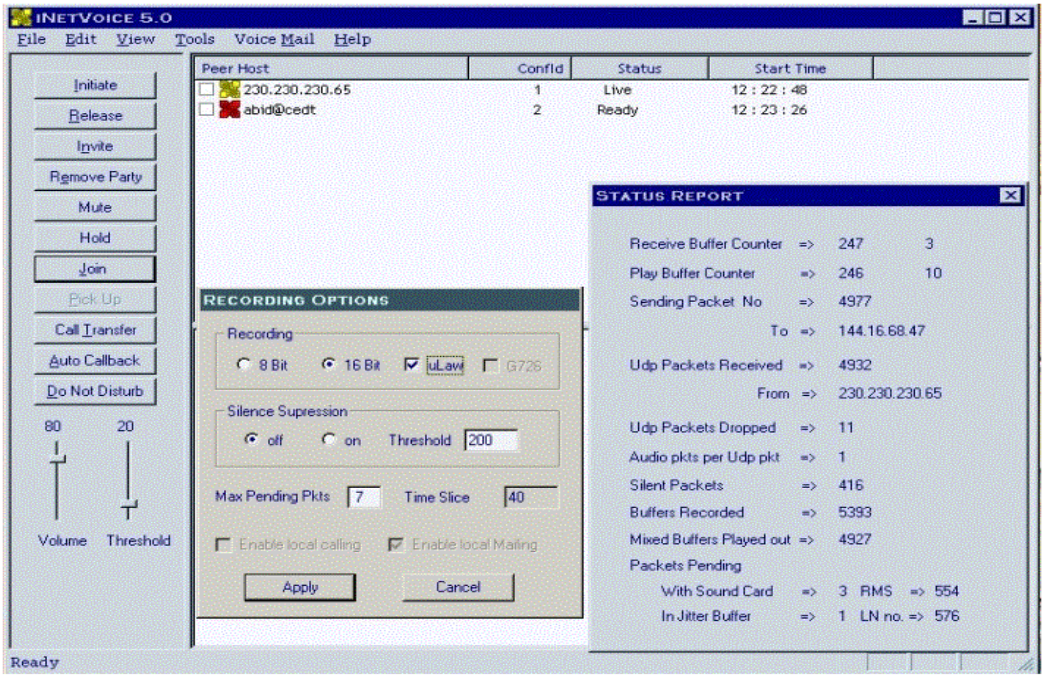


Figure A.5: **Status report and recording options at a client**

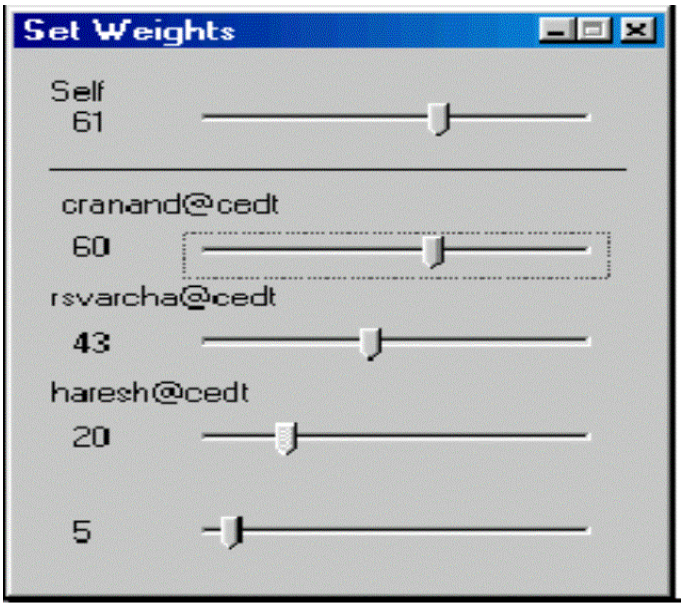


Figure A.6: **GUI for customised mixing**

References

[1] E. Schmiedeke, "IP telephony: Market demand and impact on CPE," tech. rep., Phillips InfoTech, August 1999.

[2] E. Schmiedeke, "Making IP telephony technology work for you," *Intecom, White paper*, April 2001. http://www.intecom.com/News/white_papers.htm.

[3] D. S. Isenberg, "The Dawn of the Stupid Network," *ACM netWorker*, vol. 2, no. 1, pp. 24–31, 1998.

[4] D. S. Isenberg, "The Rise of the Stupid Network," *Computer Telephony*, pp. 16–26, August 1997.

[5] A. M. Odlyzko, "The stupid network: Essential yet unattainable," *ACM netWorker*, vol. 3(4), pp. 36–37, December 1999.

[6] A. M. Odlyzko, "Smart and stupid networks: Why the Internet is like Microsoft," *ACM netWorker*, vol. 2(5), pp. 38–46, December 1998.

[7] D. F. Vrsalovic, "Intelligent, stupid and really smart networking," *ACM netWorker*, vol. 2(3), pp. 44–47, April/May 1998.

[8] J.-P. Hubaux, C. Gbaguidi, S. Koppenhoefer, and J.-Y. L. Boudec, "The impact of the Internet on telecommunication architectures," *Journal of Computer Networks*, vol. 31, pp. 257–273, 1999.

[9] J. Blickstein, "Editorial," *ACM netWorker*, vol. 2, Feb/March 1998.

[10] W. Ben-Ameur and H. Kerivin, "New economical virtual private networks," *Communications of the ACM*, vol. 46, pp. 69–73, June 2003.

[11] E. Crawly, R. Nair, R. Rajagopalan, and H. Sandick, "A framework for QoS-Based Routing in the Internet." IETF RFC, 2386, 1998.

[12] N. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. Ramakrishnan, and J. V. der Merive, "A flexible model for resource management in virtual private networks.," in *ACM SIGCOMM* (A. Press, ed.), (Cambridge, MA, New York), pp. 95–108, 1999.

[13] "Rethinking the design of the internet: The end to end arguments vs. the brave new world," *ACM Transactions on Internet Technology*, vol. 1, pp. 70–109, August 2001.

[14] W. Jiang, J. Lennox, H. Schulzrinne, and K. Singh, "Towards junking the PBX: Deploying IP telephony," (Port Jefferson, New York, USA), NOSSDAV'01, June 25-26 2001.

[15] H. Schulzrinne, "Re-Engineering the Telephone System," in *IEEE Singapore International Conference on Networks (SICON)*, (Singapore), April 1997.

[16] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: a transport protocol for real-time applications," January 1996. Request for Comments (Proposed Standard) 1889, Internet Engineering Task Force.

[17] T. J. Kostas, M. S. Borella, I. Sidhu, G. M. Schuster, J. Grabiec, and J. Mahler, "Real-Time Voice Over Packet-Switched Networks," *IEEE Network*, pp. 18–26, Jan/Feb 1998.

[18] "Webpage: http://www.idc.com."

[19] "http://www.crn.com/sections/special/bni/bni02.asp?articleid=34454."

[20] "http://www.imcca.org/marketresearch.asp."

[21] P. J. Sheppard and G. R. Walker, *Chapter: 'The future of Telephony'; Telepresence.* Kluwer Academic, 1999.

[22] L. R. Silverman, "Coming of age: Conferencing solutions cut corporate costs," White Paper: http://www.imcca.org/.

[23] M. Russ, *Chapter: 'Desktop conversations-The futeure of Multimedia Conferencing'; Telepresence.* Kluwer Academic, 1999.

[24] E. Doerry, *An Empirical Comparison of Copresent and Technologically-mediated Interaction based on Communicative Breakdown.* PhD thesis, Graduate School of the University of Oregon, 1995.

[25] A. Kumar, M. Hegde, S. V. R. Anand, B. N. Bindu, D. Thirumurthy, and A. A. Kherani, "Nonintrusive TCP connection admission control for bandwidth management of Internet access link." IEEE Communications Magazine, May 2000.

[26] E. M. Schooler, S. L. Casner, and J. Postel, "Multimedia conferencing:has it come of age?," in *Proceedings 24th Hawaii International Conference on System Sciences*, vol. 3, pp. 707–716, January 1991.

[27] M. M. Radenkovic and C. Greenhalgh, "Multi-party distributed audio service with TCP fairness," in *Proceedings of the 10th ACM International Conference on Multimedia (MM 02)*, (Juan-les-Pins, France), pp. 11–20, December 2002.

[28] M. Radenkovic and C. Greenhalgh, "Supporting collaborative audio in the internet," *ISBN 960 8052 718*, pp. 190–196, 2002.

[29] M. Radenkovic and C. Greenhalgh, "Supporting collaborative Audio in the Internet," in *Proceedings of WSEAS ICOMIV 2002*, (Skiathos, Greece), pp. 3441–3447, September 2002.

[30] J. Hindmarsh, M. Fraser, C. Heath, S. Benford, and C. Greenhalgh, "Fragmented interaction: Establishing mutual orientation in virtual environments," in *CSCW'98*, (Seattle, WA, USA), pp. 217– 226, ACM Press, November 1998.

[31] "Instruction On Demand." http://ece.iisc.ernet.in/iod/.

[32] "One-way transmission time." http://www.itu.int/itudoc/itu-t/rec/g/g114.html, 1993. ITU-T Rec. G.114.

[33] J. Bellamy, *Digital Telephony*. New york: John Wiley & Sons, Inc., 2 ed., 1991. Wiley Series in Telecommunication.

[34] R. J. D'Ortenzio, "Conferencing fundamentals for digital PABX equipments," *IEEE ICC*, pp. 29–36, 1979.

[35] L. A. Baxter, P. R. Berkowitz, C. A. Buzzard, J. J. Horenkamp, and F. E. Wyatt, "System 75: Communication and control architecture," *AT&T Technical Journal*, vol. 64, no. 1, pp. 153–173, 1985.

[36] "Chelston Call Systems Ltd, London, England." http://www.chelston.co.uk.

[37] C. Greenhalgh, S. Benford, and M. Craven, "Patterns of network and user activity in an inhabited television event," in *Proceedings of the ACM symposium on Virtual reality software and technology*, (London, United Kingdom), pp. 34–41, 1999. Virtual Reality Software and Technology.

[38] M. Radenkovic, C. Greenhalgh, and S. Benford, "Deployment issues in multi-party audio for CVEs," in *Proceedings of ACM VRST 2002*, (Hong Kong), pp. 179–185, ACM Press, Novembe 2002.

[39] M. Radenkovic, C. Greenhalgh, and S. Benford, "Scaleable and adaptable audio service for supporting collaborative work and entertainment over the Internet," in *Proceedings of International Conference on Advances in Infrastructures for e-Business, e-Education, e-Science and e-Medicine in Internet (SSGRR 2002)*, (L'Aquila, Italy), January 2002.

[40] H. A. Vin, P. V. Rangan, and S. Ramanathan, "Hierarchical conferencing architectures for inter-group multimedia collaboration," in *ACM Conference proceedings on Organizational computing systems*, (Atlanta, Georgia, United States), pp. 43–54, November 1991.

[41] S. Ramanathan, P. V. Rangan, and H. M. Vin, "Designing communication architectures for interorganizational multimedia collaboration," *Journal of Organizational Computing*, vol. 2, no. 3&4, pp. 277–302, 1992.

[42] S. Ramanathan, P. V. Rangan, H. M. Vin, and T. Kaeppner, "Optimal communication architectures for multimedia conferencing in distributed systems.," *ICDCS 1992*, pp. 46–53, 1992.

[43] E. M. Schooler, "Conferencing and collaborative computing," *Multimedia Systems*, vol. 4, no. 5, pp. 210–225, 1996.

[44] H. P. Dommel and J. Garcia-Luna-Aceves, "Networking foundations for collaborative computing at internet scope," in *International ICSC Congress on Intelligent Systems and Applications, Symposium on Interactive and Collaborative Computing (ICC'2000)*, (Wollongong, Australia), December 12 - 15 2000.

[45] H. Dommel and J. J. Garcia-Luna-Aceves, "Floor control for multimedia conferencing and collaboration," *Multimedia Systems Journal (ACM/Springer)*, vol. 5, no. 1, pp. 23–28, 1997.

[46] H. P. Dommel and J. Garcia-Luna-Aceves, "Network support for turn-taking in multimedia collaboration," in *IS&T/SPIE Symposium on Electronic Imaging: Multimedia Computing and Networking*, (San Jose, CA), February 1997.

[47] Agustn Jose Gonzalez, "A distributed audio conferencing system," July 28, 1997.

[48] Agustn Jose Gonzalez, *A Semantic-based Middleware for Multimedia Collaborative Applications*. PhD thesis, Computer Science, Old Dominion University, May 2000.

[49] C. Hendrix and W. Barfield, "Presence in virtual environments as a function of visual and auditory cues," in *Virtual Reality Annual International Symposium (VRAIS'95)*, (Research Triangle Park, North Carolina), p. 74, March 11 - 15 1995.

[50] A. Watson and A. Sasse, "The good, the bad, and the muffled: the impact of different degradations on Internet speech," in *Proceedings of MM2000*, (Los Angeles, California), 2000.

[51] E. M. Schooler, "The impact of scaling on a multimedia connection architecture," *ACM Journal of Multimedia Systems*, vol. 1, no. 1, pp. 2–9, 1993.

[52] E. M. Schooler, "Case study: Multimedia conference control in a packet-switched teleconferencing system," *Journal of Internetworking: Research and Experience*, vol. 4, no. 2, pp. 99–120, 1993.

[53] M. Handley, J. Crowcroft, C. Bormann, and J. Ott, "Very large conferences on the internet: the internet multimedia conferencing architecture," *Journal of Computer Networks*, vol. 31, pp. 191–204, 1999.

[54] A. Hac and D. chen Lu, "Architecture, design, and implementation of a multimedia conference system," *International Journal of Network Management*, vol. 7, pp. 64–83, 1997.

[55] J. Roth and C. Unger, "An extensible classification model for distribution architectures of synchronous groupware," in *Dieng R. et al. (eds): Fourth International Conference on the Design of Cooperative Systems (COOP 2000)*, (Sophia Antipolis (France)), pp. 113–127, IOS Press, May 23-26 2000.

[56] S. Deering, "Host extensions for IP multicasting." Stanford University, Stanford, CA, May 1988. RFC 1054.

[57] S. Deering, "Multicast routing in internetworks and extended LANs," *ACM SIGCOMM 88*, pp. 55–64, 1988. Host Extensions for IP Multicasting, RFC 1112 (IGMP).

[58] S. Deering, *Multicast Routing in a Datagram Internetwork*. PhD thesis, Stanford University, 1991.

[59] B. Cain, S. Deering, and A. Thyagarajan, "Internet group management protocol,version 3," October 2002. RFC 3376.

[60] K. Savetz, N. Randall, and Y. Lepage, *MBone: Multicasting Tomorrow's Internet*. IDG, April 1996. Soft copy of the book at http://www.savetz.com/mbone/.

[61] J. Rosenberg and H. Schulzrinne, "Models for multiparty conferencing in SIP," *IETF*, 2002. http://www.ietf.org/internet-drafts/draft-ietf-sipping-conferencing-models-01.txt, Internet Draft.

[62] M. Macedonia and D. Brutzman, "Mbone provides audio and vision across the internet," *IEEE Computer*, pp. 30–36, 1994.

[63] V. Jacobson, "manual pages: Lawrence berkeley laboratory," Feb. 1992. www.nrg.ee.lbl.gov/vat/.

[64] "User Handbook of Multimedia Internet Terminal (MInT)." http://www.fokus.gmd.de/research/cc/glone/products/mint/.

[65] S. McCanne and V. J. amd M. Vetterli, "Receiver-driven layered multicast," in *Proceedings of ACM SIGCOM'96*, pp. 117–130, August 1996. Palo Alto, CA.

[66] B. Furht, R. Westwater, and J. Ice, "IP simulcast: A new technique for multimedia broadcasting over the Internet," *Journal of Computing and Information Technology*, vol. 6, no. 3, pp. 245–254, 1998.

[67] S. G. Kim, Y.-M. Choi, S. T. Kim, and Y. S. Kim, "A dynamic multicast tree for loosely coupled conference," *ACM MM*, 1999.

[68] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proceedings of ACM Sigcomm 2002*, (Pittsburgh, Pennsylvania), August 2002.

[69] "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP), ITU-T Rec. G.729." http://www.itu.int/itudoc/itu-t/rec/g/g729.html, 1996.

[70] "40,32,24,16 kbit/s adaptive differential pulse code modulation (ADPCM)." ITU-T Rec. G.726; http://www.itu.int/itudoc/itu-t/rec/g/g726.html, 1996.

[71] V. Hardman, M. A. Sasse, and I. Kouvelas, "Successful multiparty audio communication over the Internet," *Communications of the ACM*, vol. 41, pp. 74–80, May 1998.

[72] M. R. Stytz, "Distributed virtual environments," *IEEE Comput. Graph. Applications.*, vol. 16, pp. 19–31, May 1996.

[73] K. S. Park and R. V. Kenyon, "Effects of network characteristics on human performance in a collaborative virtual environment," *Proc. IEEE Virtual Reality Conf.*, pp. 104–111, 1999.

[74] I. Wakeman, F. Wilson, and W. Smith, "Quality of service parameters for commercial application of video telephony," in *Human Factors in Telecommunication Symposium*, (Darmstadt, Germany), March 1993.

[75] M. Matijasevic and L. Skorin-Kapov, "Design and QoS evaluation of a multi-user virtual audio chat," in *TERENA Networking Conference*, (Limerick, Ireland), 2002.

[76] A. Watson and M. A. Sasse, "Measuring perceived quality of speech and video in multimedia conferencing applications," in *Proceedings of the sixth ACM international conference on Multimedia 1998*, (Bristol, United Kingdom), pp. 55–60.

[77] V. J. Hardman, M. A. Sasse, A. Watson, and M. Handley, "Reliable audio for use over the Internet," in *Proceedings of INET95*, (Honolulu, Oahu, Hawaii), September 1995.

[78] P. T. Brady, "Effects of transmission delay on conversational behaviour on echo-free tele-phone circuits," *Bell System Technical Journal*, pp. 115–134, 1971.

[79] S. B. Moon, J. Kurose, P. Skelly, and D. Towsley, "Correlation of packet delay and loss in the internet," tech. rep., UMASS CMPSCI, 1998.

[80] J. Pointek, F. Shull, R. Tesoriero, and A. Agrawala, "Netdyn revisited: A replicated study of network dynamics," 1996.

[81] H. Dagale, "Audio conferencing tool for Intranet," Master's thesis, Indian Institute of Science, Bangalore, India, 2001.

[82] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview," *RFC 1633*, 1994.

[83] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the differentiated services field (DS Field) in the IPv4 and IPv6 headers," *RFC 2474*, 1998.

[84] R. Braden, "Resource reservation protocol (RSVP) – version 1 functional specification," *RFC 2205(updated RFC 2750)*, 1997. Ed., L. Zhang, S. Berson, S. Herzog, S. Jamin.

[85] S. Agnihotri, "Improving quality of speech in VoIP using time-sacle modification," Master's thesis, Indian Institute of Science, Bangalore, India, 2001.

[86] S. Greenberg, "Personalizable groupware: Accomodating individual roles and group differences," in *In Proceedings of the European Conference of Computer Supported Cooperative Work (ECSCW '91)*, (Amsterdam), pp. 17–32, Kluwer Academic Press, September 24-27 1991.

[87] M. Roseman and S. Greenberg, "GROUPKIT: a groupware toolkit for building real-time conferencing applications," in *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, (Toronto, Ontario, Canada), pp. 43–50, November 1-4 1992.

[88] V. Hardman and M. Iken, "Enhanced reality audio in interactive networked environments," in *FIVE '96 Framework for Immersive Virtual Environments*, 1996. The Second International Conference of The FIVE Working Group.

[89] Kyeong-Yeol Yu and Jong-Hoon Park and Jong-Hyeong Lee, "Linear pcm signal processing for audio processing unit in multipoint video conferencing system," in *Proceedings Third IEEE Symposium on Computers and Communications*, pp. 549–553, 30 Jun-2 Jul 1998.

[90] C. Goodwin, *Notes on story structure and the organisation of participation.* Cambridge: Cambridge University Press, 1984. Structures of Social Action. Studies in conversation analysis.

[91] ITU-T Rec. H.323,
"Packet based multimedia communications systems," http://www.itu.int/itudoc/itu-t/rec/h/h323.html 1998.

[92] ITU-T Rec. H.245, "Control protocol for multimedia communication." http://www.itu.int/itudoc/itu-t/rec/h/h245.html.

[93] ITU-T Rec. T.120, "Data protocols for multimedia conferencing." http://www.itu.int/itudoc/itu-t/rec/t/t120.html.

[94] I. Dalgic and H. Fang, "Comparison of H.323 and SIP for IP telephony signaling," *Proc. of Photonics East*, 1999.

[95] H. Schulzrinne and J. Rosenberg, "A comparison of SIP and H.323 for internet telephony," *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, pp. 83–86, 1998.

[96] K. Singh and H. Schulzrinne, "Interworking between SIP/SDP and H.323," 2000. IP-TEL2000.

[97] "ITU-T Rec. T.124, generic conference control." http://www.itu.int/itudoc/itu-t/rec/t/t124.html, 1998.

[98] J. Rosenberg, H. Schulzrinne, G. camarillo, A. Jhonston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session initiation protocol," *RFC 3261, IETF*, 2002. ftp://ftp.isi.edu/in-notes/rfc3261.txt.

[99] P. Koskelainen, H. Schulzrinne, and X. Wu, "A SIP-based Conference Control Framework," in *Proceedings of NOSSDAV 2002*, May 2002.

[100] K. Singh, G. Nair, and H. Schulzrinne, "Centralized conferencing using SIP," in *Proceedings of the 2nd IP-Telephony Workshop (IPTel'2001)*, April 2001.

[101] "NetMeeting Resource Kit."
http://www.microsoft.com/windows/NetMeeting/Corp/reskit/, 2000. December.

[102] "The distributed interactive virtual environment (DIVE)." http://www.sics.se/dive/.

[103] C. Carsson and O. Hagmnd, "DIVE-A platform for multi-user virtual environments," *Computer Graphics*, vol. 17, no. 6, pp. 663–669, 1993.

[104] A. Steed and E. Frecon, "Building and supporting a large-scale collaborative virtual environment," in *Proceedings of 6th UKVRSIG,University of Salford*, pp. 59–69, 13th-15th September 1999.

[105] O. Hagsand, R. Lea, and M. Stenius, "Using spatial techniques to decrease message passing in a distributed VE," in *VRML'97*, (Monterey, CA), February 1997.

[106] C. Greenhalgh, S. Benford, A. Bullock, N. Kuijpers, and K. Donkers, "Predicting network traffic for collaborative virtual environments," in *Computer Networks and ISDN Systems*, vol. 30. First appeared in Proc. TNC'98,TERENA Networking Conference,Dresden,5-8 October 1998.

[107] V. Jacobson, "LBL network research group talks." http://www-nrg.ee.lbl.gov/nrg-talks.html.

[108] V. Jacobson, S. McCanne, and S. Floyd, "A conferencing architecture for 'light-weight sessions'," 1993. Presentation file, MICE Seminar series.

[109] "Robust Audio Tool." Networked Multimedia Research Group at University College London. http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/.

[110] A. Bullock, "A review of tools and infrastructures for distributed conferencing,seminar support and collaborative work." http://www2.inf.fh-bonn-rhein-sieg.de/ kjonas2m/lv/WS2002/pmmk1/index.shtml.

[111] S. McCanne and V. Jacobson, "vic: A flexible framework framework for packet video," *ACM Multimedia '95*, 1995. http://www-nrg.ee.lbl.gov/vic/.

[112] H. Schulzrinne, "Dynamic configuration of conferencing applications using pattern-matching multicast," in *In Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*. Lecture Notes in Computer Science (LNCS), pages 231-242, Durham, New Hampshire, Apr. 1995. Springer.

[113] S. Varakliotis, O. Hodson, and V. Hardman, "A software platform for multiway audio distribution over the internet,in audio and music technology: the challenge of creative DSP," *IEE Colloquium*, 1998. London.

[114] E. Frcon and M. Stenius, "Dive: A scaleable network architecture for distributed virtual environments," *Distributed Systems Engineering Journal (special issue on Distributed Virtual Environments)*, vol. 5, no. 3, pp. 91–100, 1998.

[115] P. V. Rangan, M. Harrick, and V. S. Ramanathan, "Architectures and algorithms for media mixing in multimedia conferences," *IEEE/ACM Transactions on Networking*, vol. 1, no. 1, 1993.

[116] H. Schulzrinne, "RFC 1890 RTP profile for audio and video conferences with minimal control." Audio-Video Transport Working Group IETF, January 1996.

[117] G. Blair and J.-B. Stefani, *Open Distributed Processing and Multimedia.* Harlow, England: Addison Wesley, 1998.

[118] N. A. Lynch, *Distributed Algorithms.* Harcourt Asia PTE Ltd. and Morgan Kaufmann, 2000.

[119] C. M. Greenhalgh, A. Bullock, J. Tromp, and S. D. Benford, *Evaluating the Network and Usability Characteristics of Virtual Reality Conferencing.* Kluwer Academic Publishers, 1999. Telepresence.

[120] G. Fairhurst. http://www.erg.abdn.ac.uk/users/gorry/course/lan-pages/csma-cd.html.

[121] B. Whetten, S. Steinberg, and D. Ferrari, "The packet starvation effect in CSMA/CD LANs and a solution," in *Proceedings 19th IEEE Local Computer Networks Conference*, (Minneapolis MN), September 1994.

[122] D. Clark and D. Tennenhouse, "Architectural considerations for a new generation of protocols," *Proc. of ACM SIGCOMM*, pp. 201–208, 1990.

[123] H. Sacks, E. Schegloff, and G. Jefferson, "A simplest systematics for the organization of turn-taking for conversations," *Language*, vol. 50, no. 4, pp. 696–735, 1974.

[124] E. Doerry, "Mosaic of Creativity," *ACM SIGCHI'95*, pp. 47–48, May 1995.

[125] K. Dingley, "Using conversational analysis to ascertain whether commercially available synchronous messengers can support real world ad hoc interactions," in *Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments*, (New Orleans, Louisiana, USA), ACM 2002 Conference on Computer Supported Co-operative Work, November 2002.

[126] Erica McAteer, "Using computer mediated conferencing (CMC)." http://www.elicit.scotcit.ac.uk/modules/cmc1/resources.htm. Universty of Glasgow.

[127] A. McKinlay, A. J. Arnott, R. Procter, and O. Masting, "A study of turn-taking in a computersupported group task," in *People and Computers VIII, Proceedings of the HCI'93 Conference*, pp. 383–394, Cambridge University Press, 1993.

[128] T. R. Shankar, M. VanKleek, A. Vicente, and B. K. Smith, "Fugue: A computer mediated conversational system that supports turn negotiation," in *Proceedings of the 33rd Hawaii International Conference on System Sciences*, pp. 3035–3044, January 2000.

[129] M. Smith, J. J. Cadiz, and B. Burkhalter, "Conversation trees and threaded chats," in *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, (Philadelphia, Pennsylvania, USA), pp. 97–105, December 2000.

[130] K. Maly, C. M. Overstreet, A. Gonzlez, M. Ireland, and N. Karunaratne, "Experiences with structured recording and replay in interactive remote instruction," in *2nd International Conference on New Learning Technologies*, (University of Berne, Switzerland.), 30-31 Augus 1999.

[131] C. Greenhalgh and S. Benford, "Massive: A collaborative virtual environment for teleconferencing," *ACM Transactions on Computer-Human Interaction*, vol. 2, no. 3, pp. 239–261, 1995.

[132] O. Hagsand, R. Lea, and M. Stenius, "Using spatial techniques to decrease message passing in a distributed VE," in *Proc. VRML'97*. The Int. Symp. on the Virtual.

[133] J. Bowers, J. Pycock, and J. O'Brien, "Talk and embodiment in collaborative virtual environments," *ACM CHI*, pp. 13–18, 1996.

[134] S. Benford, J. Bowers, L. FahlZn, C. Greenhalgh, and D. Snowdon, "User embodiment in collaborative virtual environments," *Proc. CHI'95*, 1995.

[135] C. Bormann and J. Ott, "Simple conference control protocol," in *IETF Internet Draft*, December 1996.

[136] E. A. Schegloff, *Recycled Turn Beginnings: A Precise Repair Mechanism in Conversation's Turn-Taking Organisation.* 1987. Talk and Social Organization, Cleveden Multilingual Matter.

[137] E. A. Schegloff, "Overlapping talk and the organization of turn-taking for conversation," *Language in Society*, vol. 29, no. 1, pp. 1–63, 2000.

[138] E. Schlegloff, G. Jefferson, and H. Sacks, "The preference for self-correction in the organization of repair in conversation," *Language*, vol. 53, no. 2, pp. 361–382, 1977.

[139] E. A. Schegloff, "When 'others' initiate repair," *Applied Linguistics*, vol. 21, no. 2, pp. 205–243, 2000.

[140] E. A. Schegloff, *Chapter: 'Accounts of Conduct in Interaction: Interruption, Overlap and Turntaking';Handbook of Socialogical Theory.*

[141] E. A. Schegloff, *Parties and Talking Together: Two Ways in Which Numbers Are Significant for Talk-in-Interaction.* Washington, D.C: University Press of America, 1995.

[142] E. A. Schegloff, *Turn Organization: One Intersection of Grammar and Interaction.* Cambridge: Cambridge University Press, 1996. Interaction and Grammar.

[143] E. A. Schegloff, *Third Turn Repair*, vol. 2. Amsterdam: John Benjamins, 1997. Towards a Social Science of Language: Papers in honor of William Labov.

[144] E. A. Schegloff, *Beginning in the Telephone.* Cambridge: Cambridge University Press, 2002. Perpetual Contact: Mobile communication, private talk, public performance.

[145] Selected References: Conversation analysis; http://ling.uni-konstanz.de/pages/anglistik/lehre/lehrveranstaltungen_ws98-99/convanalysis/refs-e.html.

[146] E. Isaacs and H. H. Clark, "References in conversation between experts and novices," *Journal of Experimental Psychology: General*, vol. 116, no. 1, pp. 26–37, 1987.

[147] J. A. Gonzalez and H. Abdel-Wahab, "Audio mixing for interactive multimedia communications," *JCIS'98*, pp. 217–220, 1998. Research Triangle, NC.

[148] A. C. Gyton and J. E. Hall, *Text book of Medical Physiology.* 9 ed.

[149] J. Jaffe and S. Feldstein, *Rhythms of dialogue.* New York: Academic Press, 1970.

[150] TIMIT www.ldc.upenn.edu/Catalog/LDC93S1.html. http://www.ldc.upenn.edu/readme_files/timit.readme.html.

[151] ITU, ITU-T Rec. P.800, "Methods for subjective determination of transmission quality," August 1996.

[152] Eckehard Doerry, "Personal communications," 2003.

[153] T. Johnson, "A performance comparison of fast distributed mutual exclusion algorithms," in *IPPS*, (Santa Barbara, California, USA), pp. 258–264, 1995.

[154] Helary, Jean-Michel, and M. Achour, "A O(log2n) fault-tolerant distributed mutual exclusion algorithm based on open-cube structure." RR-2041, INRIA, 1993.

[155] I. Suzuki and T. Kasami, "A distributed mutual exclusion algorithm," *ACM Transactions on Computer Systems (TOCS)*, vol. 3, no. 4, pp. 344–349, 1985.

[156] S. Lodha and A. Kshemkalyani, "A fair distributed mutual exclusion algorithm," *IEEE Trans. on Parallel and Distributed Systems*, vol. 11, no. 6, pp. 537–549, 2000.

[157] M. Mizuno, M. L. Neilsen, and R. Rao, "A token based distributed mutual exclusion algorithm based on quorum agreements," in *Proc. of 11th ICDCS'91*, pp. 361–368, May 1991.

[158] G. Button and P. Dourish, "Technomethodology: Paradoxes and possibilities," in *Proc. ACM CHI*, (Vancouver, BC Canada), pp. 19–26, 1996.

[159] C. Heath, M. Jirotka, R. Luff, and J. Hindmarch, "Unpacking collaboration: The interactional organisation of trading in a city dealing room," in *Proc. Third ECSCW'93*, (Milano, Italy), September 1993.

[160] K. Ruhleder and B. Jordan, "Co-constructing non-mutual realities: Delay-generated trouble in distributed interaction," *Journal of Computer Supported Cooperative Work*, vol. 10, no. 1, pp. 113–138, 2001.

[161] B. O'Conaill, S. Whittaker, and S. Wilbur, "Conversations over video conferences: An evaluation of the spoken aspects of video-mediated communication," *ACM Human-Computer Interaction*, vol. 8, pp. 389–428, 1993.

[162] J. Bowers, "The work to make a network work: Studying CSCW in action," in *Proceedings of the ACM CSCW '94*, (Chapel Hill, USA, ACM, NY), pp. 287–298, 1994.

[163] D. O'Shaugnessy, *Speech Communications Human and Machine*. Hyderabad: University Press, second ed., 2001.

[164] G. E. Pelton, *Voice Processing*. Singapore: McGraw-Hill International Edition, 1993.

[165] R. D. Rodman, *Computer Speech Telephony*. Boston: Artech House, 1999.

[166] G. Fant, "Some problems in voice source analysis," *J. Speech Communication*.

[167] T. F. Quatieri, *Discrete-Time Speech Signal Processing: Principles and Practice*. Prentice Hall, 2002.

[168] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*. Prentice Hall, 1978.

[169] E. Isaacs and J. C. Tang, "What video can and cannot do for collaboration," *Multimedia Systems*, vol. 2, pp. 63–73, 1994.

[170] E. Isaacs, T. Morris, and T. K. Rodriguez, "A forum for supporting interactive presentations to distributed audiences," in *ACM Proceedings CSCW*, (Chapel Hill, NC), pp. 405–416, 1994.

[171] R. W. Hamming, *Digital Filters*. Dover Publishers, 1998.

[172] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1975.

[173] Conference DSP Unified Software Design. Wipro©, 1996.

[174] T. W. Parsons, *Voice and speech processing*. McGraw-Hill Book Company, 1986.

[175] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[176] J. R. Deller, J. H. L. Hansen, and J. G. Proakis, *Discrete-Time Processing of Speech Signals*. IEEE Reprint edition, 1999.

[177] E. Zwicker, G. Flottorp, and S. S. Stevens, "Critical bandwidth and loudness summation," *J. Acoustic Society of America*, vol. 29, pp. 548–557, 1957.

[178] E. Zwicker and B. Scharf, "A model of loudness summation," *Psychological Review*, vol. 72, pp. 3–26, 1965.

[179] J. L. Verhey, *Psychoacoustics of spectro-temporal effects in masking and loudness perception*. PhD thesis, Oldenburg : Bibliotheks- und Informationssystem der University, 1998.

[180] R. Herman, M. Beckman, and K. Honda, "Subglottal pressure and final lowering in english," in *ICSLP'96*, (Philadelphia, USA), October 1996.

[181] D. rate speech coder for multimedia communications transmitting at 5.3 & 6.3 kbit/s. ITU-T Rec. G.723.1;http://www.itu.int/itudoc/itu-t/rec/g/g723.1.html, 1996.

[182] ETSI, GSM-Enhanced Full Rate Specifications 06.51, 06.60-63and 06.82.

[183] J. C. Pasquale, G. C. Polyzos, E. W. Anderson, and V. P. Kompella, "Filter propagation in dissemination trees: Trading off bandwidth and processing in continuous media networks," in *NOSSDAV'93*.

[184] J. C. Pasquale, G. C. Polyzos, E. W. Anderson, and P. Vachaspathi, "The multimedia multicast channel, internetworking: Research and experience," vol. 5, no. 4, pp. 151–162, 1994.

[185] A. Fox, S. D. Gribble, E. A. Brewer, and E. Amir, "Adapting to network and client variablility via on-demand dynamic distillation," in *In Proceedings of the Seventh International Conference on Architectural Support for Programming Languages and Operating Systems*, (Cambridge, MA), pp. 160–170, October 1996.

[186] B. Gold and N. Morgan, *Speech and Audio Signal Processing*. John Wiley & Sons, 2000.

[187] J Natvig and S. Hansen and J De Brito, "Speech processing in the pan-european digital mobile radio system(GSM)-system overview," *Globecom*, pp. 1060–1064, 1989.

[188] K. El-Maleh and P. Kabal, "Natural quality background noise coding using residual substitution," in *EUROSPEECH*, vol. 5, (Budapest), pp. 2359–2362, September 1999.

[189] A. M. Kondoz, *Digital Speech (Coding for Low Bit Rate Communications Systems).* John Wiley Publishers, 1999.

[190] J. L. Shen, J. W. Hung, and L. S. Lee, "Robust entropy-based endpoint detection for speech recognition in noisy environments," in *ICSLP'98*, (Sydney), 1998.

[191] J. Sohn, N. S. Kim, and W. Sung, "A statistical model-based voice activity detection," *IEEE signal processing letters*, vol. 6, no. 1, 1999.

[192] Y. D. Cho, K. Al-Naimi, and A. Kondoz, "Mixed decision-based noise adaption for speech enhancement," *IEEE Electronics Letters*, vol. 6, 2001. Online No. 20010368.

[193] K. El-Maleh and P. Kabal, "Comparison of voice activity detection algorithms for wireless personal communications systems," in *IEEE Canadian Conference on Electrical and Computer engineering*, pp. 470–473, May 1997.

[194] J. Sohn, N. S. Kim, and W. Sung, "A statistical model-based voice activity detection," *IEEE Signal Processing Letters*, vol. 6, no. 1, 1999.

[195] P. Pollak, P. Sovka, and J. Uhlir, "The noise suppression system for a car," in *EUROSPEECH'93*, (Berlin, Germany), pp. 1073–1076, September 1993.

[196] P. Pollak, P. Sovka, , and J. Uhlir, "Cepstral speech/pause detectors," in *Proc. of IEEE Workshop on Nonlinear Signal and Image Processing*, (Neos Marmaras, Greece), pp. 388–391, June 1995.

[197] P. Sovka and P. Pollk, "The study of speech/pause detectors for speech enhancement methods," in *EUROSPEECH'95*, (Berlin), pp. 1575–1578, 1995.

[198] E. Nemer, R. Goubran, and S. Mahmould, "Robust voice activity detection using higher-order statistics in the lpc residual domain," *IEEE Trans. Speech and Audio Proc.*, vol. 9(3), pp. 217–231, 2001.

[199] J. Zhang, W. Ward, and B. Pellom, "Phone based voice activity detection using online bayesian adaptation with conjugate normal distributions," in *ICASSP'2002*, (Orlando Florida), May 2002.

[200] R. Sarikaya and J. Hansen, "Robust detection of speech activity in the presence of noise," in *ICSLP'98*, vol. 4, (Sydney, Australia), pp. 1455–1458, December 1998.

[201] L. R. Rabiner and M. R. Sambur, "An algorithm for determining end-points of isolated utterances," *Bell Technical Journal*, pp. 297–315, 1975.

[202] P. Pollak, P. Sovka, and J. Uhlir, "Noise suppression system for a car," in *EUROSPEECH'93*, (Berlin), pp. 1073–1076, September 1993.

[203] *Multimedia Communications - Directions & Innovations.* Academic Press, Jerry D. Gibson ed., 2001.

[204] T. Enderes, S. C. Khoo, C. A. Somerville, and K. Samaras, "Impact of statistical multiplexing on voice quality in cellular networks," *Mobile Networks and Applications*, vol. 7, no. 2, pp. 153–161, 2002.

[205] http://www.nuntius.com/solutions11.html (ITU-T recommended vocoders).

[206] http://www.cisco.com/warp/public/788/voip/codec_complexity.html (G729).

[207] J. Zhang, W. Ward, and B. Pellom, "Phone based voice activity detection using online bayesian adaptation with conjugate normal distributions," in *ICASSP'02*, (Orlando Florida), May 2002.

[208] Y. D. Cho, K. Al-Naimi, and A. Kondoz, "Improved voice activity detection based on a smoothed statistical likelihood ratio," in *ICASSP'01*, 2001.

[209] http://www.espico.com/gsmv-teaklite-1.htm.

[210] http://ftp.tiaonline.org/tr-41/tr414/Public/2002-02-Vancouver/TR41.4.4. /TR41.4.4-02-02-005_Alcatel_AMR_Contribution.pdf.

[211] M. Swanson, T. Critchlo, R. Kessler, and L. Stoller, "The design of the schizophrenic workstation system," in *Proceedings of the Third Usenix Mach Symposium*, April 1993.

[212] R. Zopf, "RFC-3389 real-time transport protocol (rtp) payload for comfort noise (CN)," September 2002.

[213] http://www.ntp.org/.

[214] H. Sawashima, Y. Hori, and H. Sunahara, "Characteristics of UDP packet loss: Effect of TCP trafficc," in *Proceedings of Internet Society Conference*, 1997.

[215] J. C. Bolot, H. Crepin, and A. V. Garcia, "Analysis and control of audio packet loss in the internet," in *NOSSDAV*, 1995.

[216] J. C. Bolot, "Characterizing end-to-end packet delay and loss in the internet," *Journal of High Speed Networks*, pp. 289–298, 1993.

[217] T. T. Lee, "M/G/1/N queue with vacation time and exhaustive service discipline," *Operations Research.*

[218] O. Brun and J.-M. Garcia, "Analytical solution of finite capacity M/D/1 queues," *J. Appl. Probab*, vol. 37, no. 4, pp. 1092–1098, 2000.

[219] A. Frey and Y. Takahashi, "A note on an M/GI/1/N queue with vacation time and exhaustive service discipline," *Operations Research Letters*, 1997.

[220] K. Thompson, G. Miller, and R. Wilder, "Wide area internet traffic patterns and characteristics IEEE network," vol. 11, no. 6, pp. 10–23, 1997.

[221] R. Cole and J. Rosenbluth, "Voice over IP performance monitoring," *ACM Computer Communication Review*, vol. 31, no. 2, 2001.

[222] P. V. Rangan, H. M. Vin, , and S. Ramanathan, "Communication architectures and algorithms for media mixing in multimedia conferences," *IEEE-ACM transactions on networking*, vol. 1, no. 1, pp. 20–30, 1993.

[223] A. J. Gonzalez and H. Abdel-Wahab, "Light-weight stream synchronization framework for multimedia collaborative applications," in *ISCC'2000*, (Antibes-Juan Les Pins, France), pp. 398–403, July 2000.

[224] S. Ramanathan and P. V. Rangan, "Continuous media synchronization in distributed multimedia systems," in *3rd International Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 328–335.

[225] J. Rosenberg and H. Schulzrinne et al., "SIP: Session initiation protocol," in *RFC 3261, IETF*, June.

[226] A. B. Roach, "Session initiation protocol (SIP)-specific event notification," in *RFC 3265, IETF*, June 2002.

[227] K. Singh, G. Nair, and H. Schulzrinne, "Centralized conferencing using SIP," in *Proceedings of IPTel'2001*, April 2001.

[228] J. Rosenberg and H. Schulzrinne, "Models for multiparty conferencing in SIP," in *Internet Draft, IETF*, July 2002.

[229] M. Handley and V. Jacobson, "SDP: Session description protocol," in *RFC 2327, IETF*, April 1998.

[230] S. Guha and S. Khuller, "Greedy strikes back: Improved facility location algorithms," *Journal of Algorithms*, vol. 31, pp. 228–248, 1999.

[231] "A compendium of NP optimization problems," in *piluc@dsi.unifi.it* (P. Crescenzi and V. Kann, eds.). http://www.nada.kth.se/ viggo/problemlist/compendium.html.

[232] M. R. Garey and D. S. Johnson, "Computers and intractability." Freeman, San Francisco, 1979.

[233] C. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*. Prentice Hall, India, 2001.

[234] N. Edwardsm, *Improved approximation algorithms for the k-level facility location problem*. PhD thesis, Cornell University, Ithaca, NY, 2001.

[235] R. Neapolitan and K. Naimipour, "Foundations of algorithms." D C Heath and Company, Toronto, 1996.

[236] Martin Pal and Eva Tardos and Tom Wexler, "Facility location with hard capacities," in *Proceedings of the 42nd Annual IEEE Symposium on the Foundations of Computer Science*, 2001.

[237] M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani, "A greedy facility location algorithm analyzed using dual-fitting," in *In Proceedings of 5th International Workshop on Randomization and Approximation Techniques in Computer Science*, vol. 2129, pp. 127–137, 2001. Lecture Notes in Computer Science.

[238] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani, "Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP," *Journal of ACM*, 2002.

[239] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman, "Analysis of a local search heuristic for facility location problems," *SODA 1998*, pp. 1–10, 1998.

[240] R. Shamir and B. Dietrich, "Characterization and algorithms for greedily solvable transportation problems," *SODA, 1990.*

[241] R. A. Colburn, M. F. Cohen, S. M. Drucker, S. LeeTiernan, and A. Gupta, "Graphical enhancements for voice only conference calls," tech. rep., Microsoft Research Microsoft Corporation, October 2001. MSR-TR-2001-95.

[242] I. Kouvelas, *A Combined Network, System and User Based Approach to Improving the Quality of Multicast Audio.* PhD thesis, University College London, 1998.

[243] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing.* Prentice Hall, 2nd ed., 1999.

[244] K. Feher, *Wireless Digital Communications.* Prentice Hall India, 2001.

[245] E. M. Schooler, "A distributed architecture for multimedia conference control." ISI Research Report, ISI/RR-91-289, 1991.

List of Publication

1. R Venkatesha Prasad, Joy Kuri, H S Jamadagni, Haresh Dagale and Ravi Ravindranath, "Control Protocol for VoIP Audio Conferencing Support", *International Conference on Advanced Communication Technology (ICACT)*, Mu-Ju, South Korea, pp. 419-424, Feb. 2001.

2. R Venkatesha Prasad, H S Jamadagni, Joy Kuri, Haresh Dagale and Ravi Ravindranath, "Automatic Addition and Deletion of clients in VoIP Conferencing", *6th IEEE Symposium on Computers and Communications*, Hammamet, Tunisia, pp. 386-390, July 2001.

3. R. Venkatesha Prasad, Abhijeet Sangwan, H. S. Jamadagni, M. C. Chiranth, Rahul Sah, VishalGaurav, "Comparison of Voice Activity Detection Algorithms for VoIP", *IEEE Symposium on Computer and Communications*, Sicily, Italy, pp 530-535, July 2002.

4. R Venkatesha Prasad, H S Jamadagni, Joy Kuri, Haresh Dagale, R. S. Varchas, "A Simple Mixing Algorithm for VoIP Conference", *6th world Multiconference on Systemics, Cybernetics and Informatics*, Orlando, Florida, USA, vol. 4, pp 308-313, July 2002.

5. Abhijeet Sangwan, M. C. Chiranth, H. S. Jamadagni, Rahul Sah, R. Venkatesha Prasad, VishalGaurav, "VAD Techniques for Real-Time Speech Transmission on the Internet", *5th IEEE/IEE International Conference on High Speed Networks and Multimedia Communication*, Cheju Islands, S. Korea, pp. 42-45, July 3-5, 2002.

6. Abhijeet Sangwan, H. S. Jamadagni, Chiranth, Rahul Shah, R. Venkatesha Prasad, Vishal, "Second And Third Order Adaptible Threshold for VAD in VoIP", *Sixth International Conference on Signal Processing (ICSP '02)*, Beijing, pp. 1693-1696, August. 26-30, 2002.

7. Abhijeet Sangwan, M. C. Chiranth, R. Shah, V. Gaurav, R. Venkatesha Prasad, "Voice Activity Detection for VoIP- Time and Frequency domain Solutions", *Tenth annual IEEE Symposium on Multimedia Communications and Signal Processing*, Bangalore, pp. 20-24, November. 2001.

8. C. R. Anand, Joy Kuri, R. Venkatesha Prasad, H. S. Jamadagni, R. S. Varchas, Haresh Dagale, "An Architecture for Voice Mail Service in a VoIP LAN", *Tenth annual IEEE Symposium on Multimedia Communications and Signal Processing*, Bangalore,pp 6-10, November. 2001.

9. R. Venkatesha Prasad, Richard Hurni, H. S. Jamadagni, "A Scalable Distributed VoIP Conferencing Using SIP", *8th IEEE Symp. on Computers and Communication*, Antalaya, Turkey, July 2003.

10. R. Venkatesha Prasad, Joy Kuri, H. S. Jamadagni and Ravi Ravindranath,"A Scalable Architecture for VoIP Conferencing", to appear in *Computer, Communication and Control Technologies: CCCT '03 and The 9th. International Conference on Information Systems Analysis and Synthesis: ISAS '03*, Orlando, Florida, USA, July 2003.

11. R. Venkatesha Prasad, H. S. Jamadagni, H. N. Shankar,"On the Problem of Specifying the Number of Floors for a Voice-Only Conference on Packet Networks", to appear in *proc IEEE ITRE*, New Jersey, August. 2003.

12. R. Venkatesha Prasad, H. S. Jamadagni, Abhijeet Sangwan, M. C. Chiranth, "VAD for VoIP Using Ceptrsum", *6th IEEE/IEE HSNMC*, to be published in Lecture Notes in Computer Science (LNCS), Springer-Verlag, Estoril, Portugal July, 2003.

13. R Venkatesha Prasad, Richard Hurni, H. S. Jamadagni, "A Proposal for Distributed Conferencing on SIP using Conference Servers" to appear in *IEEE/IFIP Management of Multimedia Network and Services*, to be published in Lecture Notes in Computer Science (LNCS), Springer-Verlag, Belfast, September. 2003.

14. R. Venkatesha Prasad, Richard Hurni, H. S. Jamadagni, H. N. Shankar, "Deployment Issues of a VoIP Conferencing System in a Virtual Conferencing Environment", to appear in *ACM symposium on Virtual Reality and Software Techniques*, Osaka, Japan, October 2003.

15. R. Venkatesha Prasad, H. S. Jamadagni, "Number of Floors for a Voice-Only Conference on Packet Networks - A Conjecture", accepted for publication, *IEE Proceedings - Communications Special Issue on Internet Protocols, Technology and Applications (VoIP).*

16. R. Venkatesha Prasad, S. Vijay, H. S. Jamadagni, H. N. Shankar, "Selector Allocation for VoIP Conferencing", accepted for publication, *38th National Convention of CSI*, Delhi, India, December 2003.

17. R. Venkatesha Prasad, H. S. Jamadagni, Joy Kuri, R. S. Varchas, "A Distributed VoIP Conferencing Support Using Loudness Number", Technical Report: TR-CEDT-TE-03-01.

18. R. Venkatesha Prasad, H. S. Jamadagni, Joy Kuri, R. S. Varchas, "A Distributed VoIP Conferencing Support Using Loudness Number", revised manuscript under review, *Journal, ACM - CCR.*