User-centric Architecture for Virtual Voice-only VoIP Conferencing

R. V. Prasad[1], H. N. Shankar[2], R. S. Varchas[3], H. S. Jamadagni[3], and Przemysław Pawełczak[1]

[1] Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, 2600 GA Delft, The Netherlands
{vprasad, p.pawelczak}@ewi.tudelft.nl
[2] Faculty of Telecommunications, PES Institute of Technology, Bangalore - 560085, India
{hn.shankar}@pes.edu
[3] Esqube Communication Solutions Pvt. Ltd., Sanjay Nagar, Bangalore, India
{varchas, hsjam}@esqube.com

Abstract. Recently, porting real-time collaborative services on the Internet has been attracting much interest. Mixing packets from all clients in a conference can lead to diminished speech clarity. In this paper, we describe real-time voice conferencing service framework designed for a Virtual Conferencing Environment. We propose a system designed to accommodate a large number of participants without losing spontaneity. We surmise that change in speakers addressing the conference should not be abrupt, which transpires when the decision is exclusively based on floor control techniques leading lower quality of conference. We propose a dynamic conference control and media handling of a conference taking into account users behavior. We try to allow users to decide who should have the floor to address the conference instinctively. Highlights of this architecture are scalability, bandwidth saving and improvement in quality of the conference. The contributions of this work aid in faster and user-friendly implementations of collaborative conferencing systems on the Internet. A working prototype has been field tested on a campus wide subnet.

1 Introduction

The Internet Protocol was primarily designed for best effort data transport. However, increasingly the Internet is being used as a transport mechanism for real-time traffic. Loss and delay-constraints are characteristics that separate traditional data from voice and video applications. For "telephone conversation-type" applications there are strict requirements on parameters such as end-to-end delay and jitter. Real-time voice is an important and basic medium that draws attention for its low cost and effective connectivity across the globe, e.g., Skype or Yahoo Voice Chat. The fore runner amongst many such real-time services is virtual conferencing facility. Voice and video conferencing on the Internet are

popular [1] and they inherit several advantages [2]. While there is so much activity on the Internet catering to communications between individuals, applications for collaborative work is progressing slowly and steadily.

Over the years some important applications that made a mark in collaborative games and distributed virtual environments are Timewarp [3], MASSIVE [4], DIVE [5], JASMINE [6], and many groupware applications such as GroupWeb and GroupKit. Presently, Computer Mediated Communication (CMC) platforms are the most sought after applications for many corporate members and individuals who are working in different time zones. It not only allows integration of many data types but also saves load on the exchequer [1]. For all these developments the Internet has become a ubiquitous facilitator.

We divide many issues concerning this service into two broad groups – technical and functional. Technical issues - mainly related to the underlying network - are bandwidth, delay, delay jitter, loss concealment [7–9], conference control [10] and multicasting support [11]. We see that significantly enhanced bandwidth, powerful systems for desktop conferencing, group authoring, and distributed design are the drivers for the fundamental changes in casual and formal business interactions among participants of modern society. Functional issues pertain to operational and maintenance aspects of a CMC application concerning interactions between users and applications. It concern with behavioral aspects and other important issues such as, ease of use, comfort levels of users during usage, the effects of delay and jitter, to name a few [12, 13]. Though there are many facets to the functional issues being addressed by the multimedia community, evaluation of techniques for novel communication environments has by far fallen short of aggressive research in technological advancements. Usually it is *more-the-merrier* attitude towards bandwidth, facilities and complex control mechanisms that drives application design. Indeed, this has been aptly criticized as "keeping form the before function" [14].

Therefore the concern here, in this paper, is to build a Computer Supported Cooperative Work (CSCW) application that supports voice conference trying to provide the quality as that of a face-to-face real-life conference, acceptably closely. The specifications for this problem are definitely incomplete if they are blind to actual conduct of cultured[1] participants. In this paper we take into account our earlier study [15] on users behaviour in a conference and then try to use it to design an application for voice conferencing. We do not intend to redo the study on conversational analysis [16]; we only reinforce, by further experimentation, the advantage of such a study architecture for virtual voice conferencing on the Internet enhancing the perceived quality of the conference. We leverage the results of the analysis of our experiments of a blind conference [15] briefly to gain penetrating insights into building a virtual voice conferencing tool over VoIP. We demonstrate the significance of users behaviour in designing a scalable architecture.

[1] Participants are striving to make a sense out of their interactions during a conference.

2 Related Studies

A virtual voice conferencing over the Internet throws many challenging problems. Many of them have been addressed already by various studies [7, 8, 17, 18]. However, we feel that there are many issues and requirements that have been overlooked, which can make a virtual conferencing more realistic. We first enlist below some of the earlier studies and solutions.

Ramanathan and Venkatarangan [17] have studied in detail architectural configurations comparing many ways of building a conferencing architecture considering network delay and computational requirements for mixing. Floor control to allow a set of speakers who can address all participants is another major aspect in designing a conferencing tool. This aspect is well documented by Dommel in [19]. A tightly coupled conference control protocol belongs to the ITU-T H.323 family [20]; it is mainly for small conference sizes. An IETF draft by Rosenberg and Schulzrinne [21] discusses conferencing models with Session Initiation Protocol (SIP) [22] in the background. Implementation of centralized SIP conferencing is reported in [23]. Partial mixing is proposed by Radenkovic [18] that allows mixed and non-mixed streams to coexist allowing every ones speech to be heard. MASSIVE and DIVE [4, 5] implicitly assume that arbitrarily several participants with their aura (a virtual boundary around each participant in a virtual 3D space) colliding are able to interact with each other. They mix streams from all participants who are in the vicinity. Interactive Remote Instruction (IRI) [24] and JASMINE [6], has a tight floor control with only one participant speaking at an instance and is selected on first-come first-served basis.

Though there are many solutions listed above, we feel that there are many issues and requirements that have been overlooked, which can make a virtual conferencing more realistic and useful. While there are some extremely useful suggestions in the above proposals, they also have the following notable limitations.

1. With more than one concurrent speaker, mixing of voice streams prior to play out is essential. Thus a CSCW tool that caters voice conferencing should take care of mixing of different streams without losing the spatialism (the ability to discern who is speaking and what is spoken). Many applications avoid this by having a an explicit floor control protocol in place which allows participants to take turns one-by-one with a scheduling algorithm as in JASMINE [6], and IRI [24] however losing out on spontaneity. Floor Control [19] in a voice (video) conference enforces explicit permissions to be obtained by participants before they start speaking. It severely restricts the freedom to interrupt as in a true face-to-face conference. Thus the service becomes 'gagging' for the users [18]. Though the conference is more focused in moderated collaborative environments [6] the net effect is the same.
2. In [18], all active streams are mixed and the number of streams mixed at intermediate servers varies dynamically depending on bandwidth availability and the number of participants speaking concurrently. This would lead to fluctuations in the volume of every individual participant's voice in the mixed

stream at the terminals of listeners. This causes severe degradation in quality of conference. Primarily, mixing speech streams causes loss of spatialism of individual streams. In fact, in a voice conference streams from all participants need not be mixed; it is also undesirable. Moreover, customized mixing of streams is not feasible when the number of participants selected for mixing is varying. There must be a threshold on the number of simultaneous speakers beyond which increasing the number of speakers does not improve conference quality. Thus it is prudent to limit the number of simultaneous speakers and further to keep it a constant.

3. Introducing many intermediate mixers (Conference Servers) in stages as in [17] results in accumulation of delay at each stage. Thus they are not scalable keeping the focus on interactivity.
4. Partial mixing [18] also allows tailored mix of streams at the participants by not mixing streams when the network can support higher bandwidth. A participant receiving streams from all other participants is unwarranted [23, 25] and results in a higher demand on computational resources on the participants.
5. Clearly, if there is no limit on number of active streams, for a large conference a centralized conference [21, 23] cannot scale. Even a completely distributed conference using multicast cannot scale up since participants have to parse many streams. Moreover, traffic on participants' network increases unnecessarily.

Evidently, solutions to particular issues are provided in each of the above studies. They all address subsets of requirements for a Virtual Conference Environment (VCE) (or in general a VoIP conference/collaboration) that are listed in the next section.

3 Motivation

With the above discussion we see that though there are many extremely useful insights one can gain from the literature, there are a few important questions that have been unanswered. Recently we have proposed an empirical study [15] to answer the question "how many concurrent speakers can be allowed without degrading a real-time voice conference?" Now there is a need to study the effects of that study.

We define the floor as a token that permits a participant to address the conference. The number of floors, N, i.e., allowing N concurrent speakers out of M participants, is an important parameter in the design of conferencing applications since it influences many aspects of collaborative tools. It governs how a floor control needs to be designed and it also influences the design of applications if there are more than one floors. Standards for such collaborative application development mention general guidelines for architecture designs, protocol, etc. but do not explicate on the value of N to be chosen. IETF's SIP [22] or ITU-T's H.323 [20] does not specify N or even a bound on N. While H.323 leaves it to collaborative tool designers, SIP doesn't even admit an instance where it may

be useful to find it. Architectures designed independent of fixing and managing N floors necessarily serve little purpose.

Next, if we assume that we have some how fixed N, how to share these floors amongst the participants is an important question that needs to be answered. What will be the architecture to provide such a service? Whether the quality of the conferencing from the participants' view would be any better? Thus we start with these main requirements:

1. Voice traffic on the network should be as low as possible.
2. Number of participants who are allowed to speak concurrently should give the feeling of a real-life face to face conference.
3. Each participant should get the same mix of voice streams.
4. Scalability must ensure that a large number of participants dispersed over a wide geographical area are easily handled.
5. Users should be able to have a tailored mix of streams.
6. Availability of multicasting in the network should not be assumed; architecture should be able to use multicast if available, say IPv6, Overlay networks [26], etc.

Therefore, there is a need to address and understand voice conferencing from a holistic perspective, i.e., users' behavior and the network support in the present day Internet. Clearly we need to understand human interactions in a conference before we set forth towards designing a conferencing tool. In the next section we briefly present our empirical study to fix the number of floors and then we proceed to propose our architecture.

4 Number of Floors: How small is enough?

Turn-taking is one of the most important aspects in any social interaction with which humans interact with others meaningfully. Using turn-taking mechanisms they organize to take turns to speak to others in group conversations this has been thoroughly studied [16]. In the present context of the virtual voice-only conferencing turn-taking is nothing but accessing the floor. As we have discussed in earlier sections we do not intend to explicitly control the user behavior by notifying who should address at an instance. We strive only to facilitate instinctive conversation with a natural turn-taking in a CSCW tool.

Here we briefly present the results of many blind conferences to mimic a voice-only conference on the Internet, so that, we will be in a position to fix N to proceed further. Details of the experiments and statistical study are provided in [15]. We present here the essence of the study. In a conference hall we allowed all the participants to interact with each other without having the visual cue. We observed how participants behave in a conference, which to a certain extent mimicked a CSCW environment. In principle we specifically wanted to simulate a conference that inherently encouraged many concurrent speakers contributing to more interruptions and more number of turn-taking without any inhibition.

Table 1. Mean μ and standard deviation σ of duration D_j when j concurrent participants are speaking (in seconds).

Duration	μ	σ
D_0	66.96	21.45
D_1	199.94	41.98
D_2	17.98	4.24
D_3	2.16	0.63
$D_{j\geqslant 4}$	2.02	2.35

We then measured the duration of the activity of their speech and measured the duration of overlapped speech. We used Goldwave, a visual tool, to see the waveforms of recorded speech. Durations were fixed by repeated listening to smaller segments of speech waveform and correlating it with the visual aid. Here D_j represents the total duration for which there were j concurrent speakers in a conference. Table 1 summarizes the mean and deviation. We have compiled the duration where more than three participants were speaking in the row $D_{j\geqslant 4}$. It may be noted that conferences were held for its full length duration ranging from 15-30 minutes. We have randomly taken three minutes samples from the middle of the conversations allowing for grounding. The normalized duration of each of the 10 samples is 287 seconds.

4.1 Some Observations

We have excluded all the details of the sample conferences. However, we present the important observations here. For a detailed account of experiments and analysis we refer to [15].

1. Mean duration of silence is 23% of total discussion time since participants took time to co-ordinate themselves, which may increase with a CSCW tool possibly to allow for catching up with network and packetisation delays. It must be emphasized, however, that such result mainly depends on participants.
2. Predominantly, 69.5% of the conversation time, only one participant was speaking. It implies that the turn-taking were effective in these experiments and any intelligible conference will have to have a repair mechanism so that the turn-taking effectively allows one person to address the gathering.
3. Two simultaneous streams (two concurrent speeches) amount to nearly the rest of the time, 6% of the duration.
4. A third concurrent speaker comes into picture for no more than 0.7% of the conversations. Moreover this phenomenon was observed when two participants got into fierce argument or when participants tried to get the attention of others during a heated debate.
5. More than three simultaneous streams are effectively nonexistent, again 0.7%. This is pivotal in choosing the number of floors N.

6. Speech intelligibility deteriorated very sharply with the third participant getting into addressing mode in any conversation. Naturally, this has led to one or more of them retracting quickly.

4.2 Fixing N

It is clearly seen from Table 1 that mostly one or two participants were talking during these experiments even without explicit floor control. Three or more participants occupy statistically less than 4 seconds of duration. Thus we may conjecture that $N = 2$ is sufficient. However, during a voice-only conference CSCW setup and when participants are geographically apart, different delays between them may result in more collisions. To allow the feedback of these collisions that are detrimental to facilitate turn-taking as well as to allow breaking potentially long dialogues, we increase the threshold of N from two to three. Hence we conjecture that in a blind (voice-only) conference three simultaneous speakers are sufficient for interactivity[2].

5 Architecture

Now with the above result specifying a bound on the number of floors to achieve spontaneous speech, we need to put the frame work that can bring this together. We propose an architecture that is dependent on the above result.

For example, in Fig. 1, there are three major locations, each of which has an arbitrary number of participants. There are two parts in the software of our framework. The 'front-end' consists of the 'client' application that runs on participants' computers. 'Back-end' is provided by other server programs that facilitate conferencing. We seek to identify two servers here as Call Processor (CP) and Selector. Our proposed architecture for conferencing is shown in Fig. 1 and 1. Functionally, CP does facilitate the exchange of control messages. Selectors facilitate the switching and selection of voice packets from various domains. Selectors and Clients are required to register with the CP. CP functionality can also be distributed, though for simplicity of presentation, we consider here one CP for a conference. The CP implements all the control messaging required for call/conference set up. Fig. 2 shows voice flow (path) between clients and selectors and between the selectors. The CP assigns clients to particular selectors and forms selector groups as in 2. The CP informs the corresponding selector when a client joins the conference. Then selector prepares itself to serve the client that

[2] We have presented here the important aspect of any conference, i.e., interactivity and allowing impromptu speech. We try to allow 'enough' number of concurrent speakers in a conference such that the participants feel that they are able to converse freely without any inhibitions. The idea here is to allow participants themselves to manage the turn-taking rather than an algorithm or a controller which would be blind to the actual conversation. This is in contrast with many conferencing solutions where it is assumed that arbitrarily many speakers are allowed [17, 18] or a tight floor control is used [19].

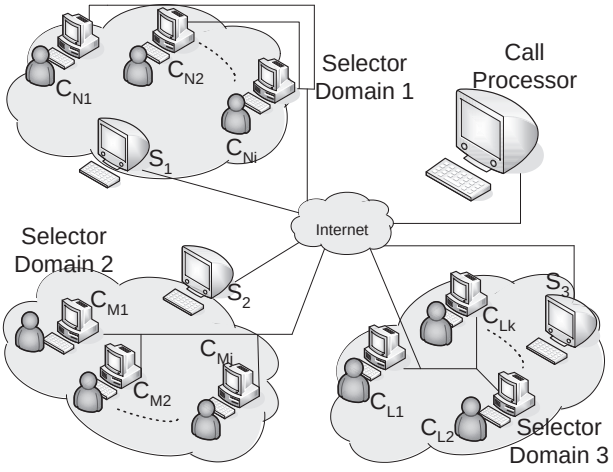

Fig. 1. Control Path in the Architecture.

has joined the conference. If the arrival of a new client brings up a new selector, and in turn a new selector group, then the CP informs all the existing selectors. We note here that the problem of assigning clients to selectors to reduce network traffic and computation load on selectors is found to be an NP-hard problem. Some heuristic algorithms can be used to solve this[27].

5.1 Working of a Selector

Functional diagram of a Selector is shown in Fig. 3. Each selector, for each mixing interval, T miliseconds period, it selects 'best' N participants from its group of clients and sends it to other selectors. Further for the same time interval it receives N packets from other selectors. For example, Selector 1 serves Clients 1 to 10 (M_1). For each mixing interval, Selector 1 chooses the best N voice packets out of $M_1 > N$ it may possibly receive, and sends these to Selectors 2 and 3. The set of packets sent is denoted as "ToOtherSelectors". In the same mixing interval, it also receives the best N voice packets (out of possibly $M_2 > N$) from Selector 2, and the best N (out of possibly $M_3 > N$) from Selector 3. The set of packets received from all other selectors is grouped and is denoted as "FromOtherSelectors". Finally, it selects the best N packets from the set ToOtherSelectors $\cup$ FromOtherSelectors to form the set $\mathbb{S}$ and sends packets in $\mathbb{S}$ to its own group. It can be seen that $\mathbb{S}$ will be the same at all selectors.

Selectors can multicast if the underlying network allows multicasting. In case multicast is not available then unicast is used between them similar to Application Level Multicasting (ALM) [26]. As most LAN technologies support multicasting, selectors normally talk to their clients on multicast. But in cases where clients cannot receive voice packets on multicast, selectors have to send unicast

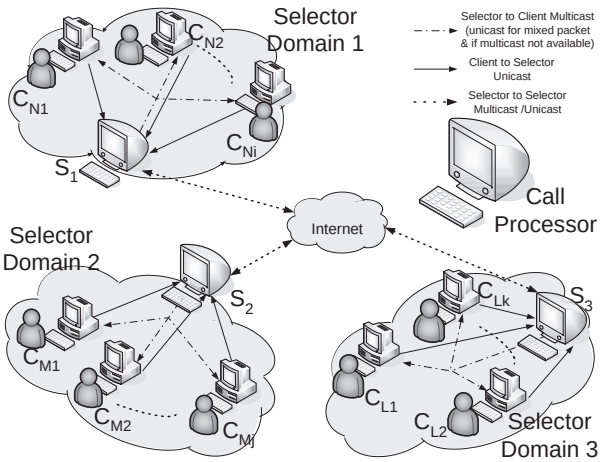

Fig. 2. Voice Path in the Architecture.

packets to those clients. Selectors also convert voice stream formats if necessary. Finally, selectors send all N packets from the set $\mathbb{S}$ to all clients in its domain along with their source identification (ID).

Each client mixes the received packets with adjustable weights and then mixed voice stream is played out. Mixing the selected packets (as in [17, 18]) before exchanging with other selectors - though it results in reduced network traffic - may consequently mix many streams (more than N) when there are several selectors and in turn decrease the quality of the conference [25]. Now the question is how to select N participants dynamically to allow faster feedback for turn-taking. The "best" N packets are decided based on the Activity Index given in next section.

5.2 Activity Index

Speaker change should be effected dynamically, smoothly and in real-time without participants being able to notice it. Thus selectors need a measure to pick the best N speakers. For this we use the following attributes of speakers:

1. current loudness of a speech,
2. loudness of a speech in the recent past duration,
3. level of activity in a reasonably past duration.

Therefore activity index is defined as a function of the current energy of the speech stream, energy in the past and level of activity for some duration in immediate past. A persistent participant would also get a chance even the loudness is lower. A complete description can be found in [25], where it is called Loudness Number. It is identified that there are some participants who shy away from

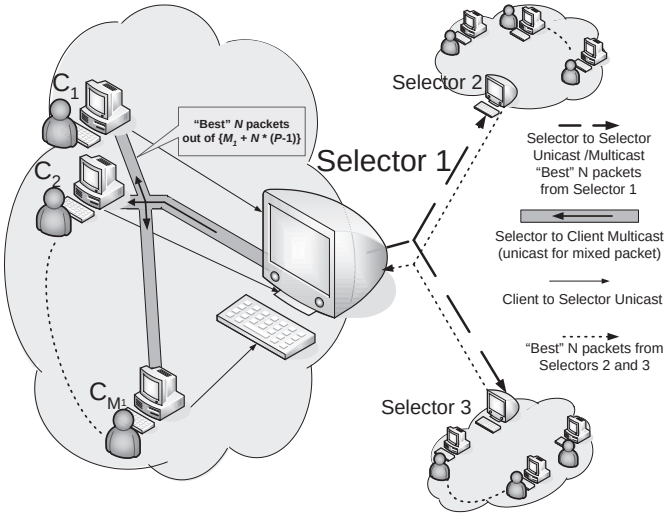

Fig. 3. Functional diagram of a Selector.

competing for a floor. We can always provide an option for participants by reserving one floor out of three, which may be allotted using techniques proposed in [6].

5.3 Some Discussion on Scalability

First, we consider non-availability of multicasting between selectors. This assumption is valid in the present day Internet as the network is not fully multicast enabled [21] and MBone covered only a few networks. The order of increase in the network traffic is as shown in Fig. 5.3a (upper most curve). Without any server handling the media, the number of packets in the network is $O(M^2)$, where M is the total number of clients in the conference. For centralized speech server scheme, i.e., clients contacting a server and server picks $N = 3$ packets and sends it back to all, it is $O(M)$. With the proposed approach, selectors handling the clients in their respective groups, for clients in P selector groups, it is $O(P^2)$, which can be at most $O(M)$. The upper bound on the number of groups, P, can be found such that the traffic because of grouping will not exceed that of the centralized scheme. Thus P is given by

$$P = \left\lfloor \frac{1}{2}\left(1 + \sqrt{1 + \left(\frac{M48}{9}\right)}\right)\right\rfloor.$$

All the above calculations assume that the clients are distributed almost equally amongst the selectors and selector picks three best clients for mixing, i.e., $N = 3$ [15, 25]. Here P is $O(\sqrt{M})$. However practically, P would be far

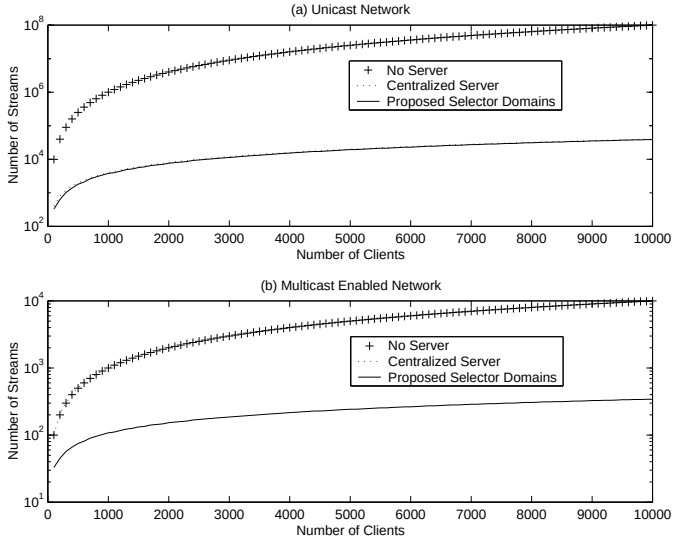

Fig. 4. (a) Traffic on the Network for Conferencing with Unicasting, multicasting not available; (b) Traffic on the Network for Conferencing with Multicast enabled Network.

less, resulting in more saving in the network traffic. Fig. 5.3a shows almost similar curves for centralized server and proposed method of selector groups but the values for latter is for an upper bound value of P (worst case). Moreover the centralized server cannot scale up arbitrarily. Secondly, assuming that the network allows multicast, Fig. 5.3b shows the similar set of results with more reduction in number of packets transported. With and without centralized server for speech packets the number of packets in the network is $O(M)$ in this case. With P selector groups handling the conference, the network traffic is $O(P)$. This shows a significant reduction in network traffic. Only the packets exchanged between selectors are considered here and packets from clients to selector are not taken into account as clients in the neighborhood (LAN or very near in terms of hop counts) are grouped together usually and this traffic can be considered almost local. Thus grouping of clients would result in a significant saving in the bandwidth in both the above cases, i.e., with and without multicast facility enabled in the network.

Simulations study in [28] showed that if number of participants is less, then there exists a break-even point below which multicasting may not be cost effective (taking into account hop count) depending on bandwidth requirement of the application. Further, possibly wide distribution of participants in the network use of ALM [26] like the one presented here is more useful. However there is a catch; traffic in ALM can be as bad as unicast [28] without proper assignment of clients [27]. In the absence of multicasting support at the network layer, the solution proposed here is the best possible keeping in mind interactivity. Recall

that selectors send best N streams in each interval to its counterparts (Section 5.1). We can harness the slow varying nature of Activity Index to reduce the total number of streams exchanged between selectors to almost N. The idea is to keep track of the least of the best N streams at the selectors in the previous interval and filter out all the streams when participants in that selector group are not active. That is when not many participants are talking concurrently, which is usually the case. Thus bandwidth can be saved significantly.

6 Conclusions

We have proposed an architecture that harnesses users' behaviour so that a virtual voice-only conference application on the Internet renders almost a face-face meeting feeling. We have taken into account the functional as well as networking aspects. We implemented a proof-of-concept test-bed and examined our overall proposal on a campus wide network, with $N = \{2, 3, 4\}$. With our limited testing the quality of the conference was better than controlled conferences. Moreover, the degree of interactivity was higher compared to conference with floor control. The main advantage of this architecture is the reduction in traffic on the network, see Fig. 5.3 as each selector filters out many clients by selecting only N. This architecture can easily be deployed in the existing Internet as selectors may talk to each other on unicast or multicast depending on the network support. Frequently, the argument is that a conference is well behaved and no more than one speaker addresses the conference at any instant of time. However in this set up no such assumption leading to hindrance in the interactivity amongst the conference participants is made.

A persistent speaker is favored to address the conference. This set up models very closely, a typical face-to-face conference. Significantly, there is no need for any explicit control messages for floor access (controller or management protocol like CCCP [10]). Since we do not mix at the Selectors users can set weights for mixing (customized mixing) bestows an opportunity for bolstering the voice of the client whom a participant prefers to hear most clearly. Further, since each stream sees only two selectors the delay will be less compared to the mixing architectures proposed earlier [17, 18]. Thus this setup helps in improving the "Quality of Experience" for participants.

References

1. (2005) Coming of age: Conferencing solutions cut corporate costs. [Online]. Available: http://www.imcca.org/wpcomingofage.asp
2. M. Decina and V. Trecordi, "Voice over internet protocol and human assisted e-commerce," *IEEE Commun. Mag.*, vol. 37, no. 9, pp. 64–67, Sept. 1999.
3. W. K. Edwards and E. D. Mynatt, "Timewarp: techniques for autonomous collaboration," in *Proc. SIGCHI Conference on Human Factors in Computing Systems*, Atlanta, GA, Mar.22–27, 1997, pp. 218–255.

4. C. Greenhalgh and S. Benford, "MASSIVE: A collaborative virtual environment for teleconferencing," *ACM Trans. on Comp.-Hum. Interaction*, vol. 2, pp. 239–261, Sept. 1995.
5. E. Frcon, C. Greenhalgh, and M. Stenius, "The DiveBone: an applicationlevel network architecture for internet-based cves," in *Proc. ACM Symposium on Virtual Reality Software and Technology*, London, UK, Dec. 1999, pp. 20–22.
6. S. Shirmohammadi, A. El-Saddik, N. D. Georganas, and R. Steinmetz, "JASMINE: A java tool for multimedia collaboration on the internet," *J. Multim. Tools and App.*, vol. 19, pp. 5–28, Jan. 2003.
7. J.-C. Bolot and A. Vega-Garca, "Control mechanisms for packet audio in the internet," in *Proc. IEEE INFOCOM'96*, San Francisco, CA, Mar.24–28, 1996, pp. 232–239.
8. S. B. Moon, J. Kurose, and D. Towsley, "Packet audio playout delay adjustment: performance bounds and algorithms," *Multimedia Systems*, vol. 6, no. 1, pp. 17–28, Feb. 1998.
9. Kitawaki and K. Itoh, "Pure delay effects on speech quality in telecommunications," *IEEE J. Select. Areas Commun.*, vol. 9, no. 4, pp. 17–28, May 1991.
10. M. Handley, I. Wakeman, and J. Crowcroft, "The conference control channel protocol (CCCP): A scalable base for building conference control applications," in *Proc. ACM SIGCOMM*, Cambridge, MA, 1995, pp. 275–287.
11. D. Thaler, M. Handley, and D. Estrin, "RFC 2908: The internet multicast address allocation architecture," Sept. 2000.
12. T. Henderson, "Latency and user behaviour on a multiplayer game server," in *Proc. 3rd International COST264 Workshop on Networked Group Communication*, London, UK, Nov.7–9,, pp. 1–13.
13. P. T. Brady, "Effects of transmission delay on conversational behaviour on echo-free telephone circuits," *Bell Syst. Tech. J.*, vol. 9, no. 4, pp. 115–134, Jan. 1971.
14. E. Doerry, "Evaluating distributed environments based on communicative efficacy," in *Proc. Conference on Human Factors in Computing Systems*, Denver, CO, May7–11,, pp. 47–48.
15. R. Prasad, H. N. Shankar, H. S. Jamadagni, and P. Pawełczak, "Fixing number of floors for virtual voice-only conference — an empirical study," in *Proc. 7th IEEE International Symposium on Multimedia*, Irvine, CA, Dec.12–14 2005, pp. 120–127.
16. E. S. H. Sacks and G. Jefferson, "A simplest systematics for the organization of turn-taking for conversations," *Language*, vol. 50, pp. 696–735, Dec. 1974.
17. S. Ramanathan, P. V. Rangan, and H. M. Vin, "Designing communication architectures for interorganizational multimedia collaboration," *J. Organiz. Comput.*, vol. 2, no. 3, pp. 277–302, Dec. 1992.
18. M. Radenkovic and C. Greenhalgh, "Multi-party distributed audio service with TCP fairness," in *Proc. 10th ACM International Conference on Multimedia*, Juan-les-Pins, France, Dec. 2002, pp. 11–20.
19. H. P. Dommel and J. Garcia-Luna-Aceves, "Floor control for multimedia conferencing and collaboration," *Multim. Syst.*, vol. 5, no. 1, pp. 23–38, Jan. 1997.
20. "ITU-T rec. H.323: Packet based multimedia communications systems," 1998. [Online]. Available: http://www.itu.int/itudoc/itu-t/rec/h/h323.html
21. J. Rosenberg and H. Schulzrinne, "RFC 4376: Models for multy party conferencing in sip," July 2002.
22. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "RFC 3261: Session initiation protocol," June 2002.
23. K. Singh, G. Nair, and H. Schulzrinne, "Centralized conferencing using sip," in *Proc. 2nd IP-Telephony Workshop (IPTel'2001)*, Juan-les-Pins, France, Apr. 2001.

24. K. Maly, C. M. Overstreet, A. Gonzlez, M. Ireland, and N. Karunaratne, "Experiences with structured recording and replay in interactive remote instruction," in *Proc. 2nd International Conference on New Learning Technologies*, University of Berne, Switzerland, Aug.30–31, 1999. [Online]. Available: http://www.cs.odu.edu/~iri-h/publications/conference_at_Berne.pdf
25. R. Prasad, H. Jamadagni, and H. N. Shankar, "Number of floors for a voice-only conference on packet networks — a conjecture," *IEE Communications Proceedings*, vol. 151, no. 3, pp. 287–291, June 2004.
26. M. Castro, P. D, A.-M. Kermarrec, and A. Rowstron, "Scribe: A large-scale and decentralized application-level multicast infrastructure," *IEEE J. Select. Areas Commun.*, vol. 20, no. 8, Oct. 2002.
27. R. Prasad, H. Shankar, H. Jamadagni, M. Rohith, and S. Vijay, "Heuristic algorithms for server allocation in distributed voip conferencing," in *Proc. IEEE Symposium on Computers and Communications*, Cartegena, Spain, Aug.30–31 2005.
28. M. Janic, "Multicast in network and application layer," PhD Dissertation, Delft University of Technology, Delft, the Netherlands, Oct. 2005.