foo

# Experience with Large Videoconferences on Xunet II

S. Keshav <keshav@research.att.com>

## Abstract

*We have been conducting bi-weekly videoconferences with up to twelve cameras and twenty speakers over Xunet II, a high-speed wide-area ATM network. We found that current Internet technology, though promising, is not adequate for production-quality videoconferences with multiple active participants. In this paper, we describe our experience with off-the-shelf Internet packages (*`nv` *and* `vat`*), and make recommendations for future work in the area.*

## I. Introduction

XUNET II is an experimental wide-area ATM network that serves as a testbed for research on data networking (Figure 1). Long distance transmission is based on DS3 trunks operating at 45 Mbps and optically amplified lines operating at 622 Mbps. Local distribution uses FDDI rings. Currently, the primary service offered over Xunet II is a fast IP interconnect between FDDI rings. We are able to achieve a peak user-to-user throughput of 38Mbps [2]

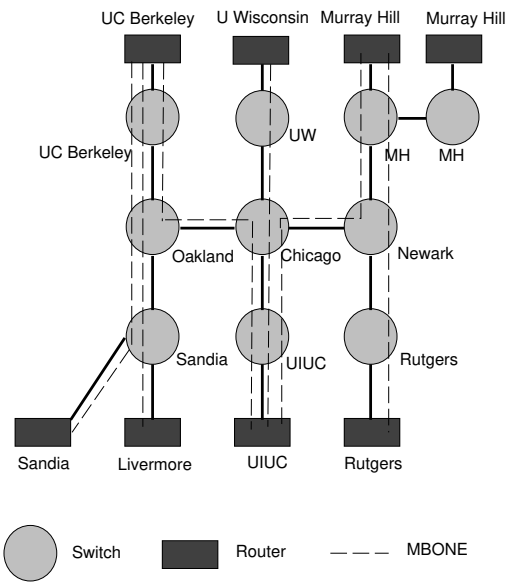


*Figure 1: Xunet configuration, showing MBONE*

This large available bandwidth makes Xunet II an ideal testbed for experiments on multimedia communication. We have been carrying video and audio traffic over Xunet since August 1993. Since Xunet offers seamless IP connectivity, we were able to use some common IP-multicast based tools with no further modification. Thus, our experience is a glimpse at how existing tools would work were they to be used over a faster Internet.

When we started, we hoped that the tools would enable production-quality videoconferencing. Unfortunately, we discovered that several problems need to be resolved before this is possible. The aim of this paper is to report these problems, thus encouraging research into solving them. We also address some myths about videoconferencing in future integrated networks.

In Section II, we discuss the tools we used. We outline some problems with the tools in Section III, and address some myths about videoconferencing in Section IV. Section V discusses areas for future work.

## II. The tools

We set up the videoconferences using the freely available `sd`, `vat` and `nv` tools [3, 5, 6]. These tools are designed to work over the MBONE (multicast backbone), a virtual private network over IP. Figure 1 shows the MBONE subtree that we used in our experiments.

`sd` is a session directory program. It supports the notion of a ‘‘session’’ which is a time over which a particular multicast address is active. Any user may create a multicast session and name it. `sd` propagates information about the new session through the network, allowing other users to join it. `sd` allows each session to be associated with a multicast- based tool, so that users can launch the appropriate tool at their local site. Typically, users start up `sd` and leave it active in the background. When new sessions are started, they appear in the current listing, and can then be joined.

`vat` is an audioconferencing tool. It uses IP multicast to distribute a user’s audio signal to multiple recipients. When the echo-suppression mode is turned on, as is typically the case, at a given moment only one person in the session may talk. `vat` creates a window that displays the names of all active participants. The name of the current speaker is highlighted, and it is possible to completely mute one’s transmission, or choose to ignore a signal from one or more transmitters.

`nv` (network video) is a program that is analogous to `vat`. It allows a user to receive video from one or more transmitters, and transmit software-compressed video to multiple receivers. A user can

limit the maximum transmission bandwidth using a slider (see Figure 2).

*Figure 2: Screen dump of a videoconference*

The three programs together provide an environment not only for one-to-many lecture or broadcast type sessions, but also for many-to-many videoconferencing sessions. In a typical use of these tools, a user would start up `sd`, and then join in one or more `vat` or `nv` conferences in progress. He or she might also create a new `vat` or `nv` session that other users can join. A user may participate in multiple sessions simultaneously (though may receive or transmit on only one `vat` session at a time).

## III. Problems with IP Multicast tools

We have been holding regular videoconferences since September 1993 to encourage co-operative research in Xunet II and to gain experience in using videoconferencing technology. Usually, large multicast sessions using `vat` and `nv` have a single speaker, or a small number of speakers, and a few active video feeds. Our videoconferences differ from typical usage in that the number of potential speakers, as well as the number of video sessions, is much larger. In the largest conference, we had twelve active video feeds and twenty-five participants at six geographically distributed sites (Figure 2).

During our videoconferences, we experienced several problems. Some of them were a result of the large size of the videoconference, while others were independent of the conference size. We discuss them in detail below.

### III.A. Poor video quality

Good quality video usually requires a frame rate of 15-30 frames/sec. The frame rate that we achieved using `nv` was between 0.1 and 1.5 frames/sec. This led to a jerky picture and loss of synchronization between lip movement and speech. Thus, while we could get a rough idea of what a participant was doing, rapid movements were lost.

We initially suspected that this was because of insufficient network bandwidth to support multiple active video feeds. However, the frame rate stayed the same even with a single video feed, and increased when the transmitting workstation was upgraded from an SGI Indigo R3000 to an SGI Indigo R4000 machine. So, the reason for the low frame rate was probably the rate at which `nv` captured images from the digitizer and compressed them. With a faster compression algorithm, or with hardware-assisted compression, it could achieve faster rates.

The slow frame update rate led to two problems with individual frames being displayed. First, some frames had areas where no update had been performed (such as in the `nv` image at left center in Figure 2). This is because `nv` updates an area when it notices a change, and there had been no movement in that area. Second, consecutive frames were sometimes simultaneously visible (as in the top left `nv` image in Figure 2). This was the result of a sudden movement. The second problem can be solved by double buffering, with a complete frame being displayed while the next frame is updated off-line.

Another unexpected problem with video quality was caused by inadequate lighting in many graduate student offices. These offices were so dimly lit that it was hard to make out the identity of the speaker. One hopes that future generations of graduate students will have better lit offices, if only for their advisors to keep an eye on them.

### III.B. Complex user interface design

The user interfaces of the tools were designed with experts in mind, which is acceptable for a research prototype. But, if they are to get widespread acceptance, participating in a videoconference should be as easy as talking on the telephone. This was not the case. For example, `vat` has slider knobs to adjust the microphone and speaker gain. It was hard to guess what the right gain setting ought to be (there is an automatic gain control setting, but in the version we used, this did not

seem to work correctly). In general, the tools need much work to make them easier to use.

### III.C. Lack of conference control

We quickly realized that a large conference needs a moderator to decide who speaks next. In our first (unmoderated) conference we had situations where there was complete silence for several tens of seconds, since no one wanted to talk over someone else. In face-to-face meetings, a potential speaker probably senses the ‘‘body language’’ of the other participants and waits till there is a clear pause in the conversation. With low frame rates, it was hard to sense that someone else wanted to speak, leading sometimes to long timeouts and at other times to collisions between speakers.

Things were better with a moderator. A moderator can guide a conference by confirming that no one wishes to speak on a particular topic or break up any collisions. It would also help to have an explicit conference control tool that would allow participants to signal their interest in speaking (similar to raising ones hand in class).

### III.D *Push to talk* problem

A combination of two problems made it inconvenient to use `vat`. First, with echo-suppression mode turned on, `vat` allowed only one speaker at a time. If someone spoke when another speaker was active, their signal was lost. Second, `vat` was sensitive to background noise, that is, it confused background noise with speech. Together, this meant that if `vat` was used in a somewhat noisy room, then it assumed that the person in the noisy room was constantly speaking, and did not allow anyone else to speak. This was particularly annoying when a participant newly joining an audio session would dominate it with their computer’s fan noise. Since they could not hear the other speakers, it was hard to instruct the new participant to mute their microphone.

Our solution to this problem was to mandate that all sites use `vat`’s ‘‘push to talk’’ feature. When this feature is selected, a user must push a mouse button while talking. Even if there is background noise, if the button is not pushed, the signal is ignored. This feature worked well if each microphone was used by a single speaker. But, when a single workstation was shared by multiple speakers, it was irritating to share a single mouse among many speakers.

### III.E. Limited screen space

It was impossible to keep more than about five `nv` windows on screen at the same time. So, we kept only some windows open, and manually switched between windows as the speakers changed.

In the recently developed `vic` video tool [9] (yet to be publicly released, at the time of writing), this problem is solved using a voice-based switching algorithm. A single video window is kept open, and it always shows the person who is speaking. So, a listener does not have to manually open and close windows. Interestingly enough, the same technique was used by the AT&T Picturephone system around 1968.

### III.F. Poor picture resolution

The picture resolution of `nv` was poor. In particular, it was not possible to read text that was held up to a camera, or even read viewgraphs sent from poorly lit room. The `wb` tool, a distributed whiteboard [7], addresses this issue by allowing participants to share a postscript image of the viewgraphs. However, we found that this tool took up most of the screen space, and much of the CPU. Since `wb`’s CPU-intensive behavior sometimes broke up the audio, we did not use it regularly.

### III.G. Slow `sd` startup

The time between starting up `sd` and getting the latest session status is variable, and sometimes long (up to several minutes). Thus, if an Xunet videoconference session was missing in the list of current sessions, we did not know whether the session had not been created, or whether it had not been updated from a remote location. This sometimes led to the creation of multiple sessions, further confusing users. This is another area where a conference control tool may help coordinate participants.

### III.H. Discussion

Despite the faults cataloged above, the Internet tools show great promise for the future. We had good quality audio once the gain was properly set. As hardware for video compression becomes available, it is likely that the video quality will also improve. We were certainly able to use the videoconferencing environment to do useful work, albeit after some fine tuning.

In the rest of the paper, we discuss some myths about videoconferencing, and then describe several ways in which the tools could be improved.

## IV. Myths about videoconferencing

Videoconferencing has been thought to be a ‘‘killer application’’, an application that will drive the future of multimedia networking. Judging from our experience, some common intuitions about this application might just be myths.

### IV.A. Need for special purpose systems

All the Internet tools run over Unix operating systems and general purpose workstations. While

the resulting quality of the system is not excellent, this shows that special purpose systems are not a prerequisite for video and good quality audio. We do believe that better scheduling of realtime activities and hardware assistance for video compression would improve the quality of the system. However, with increases in the speed of general purpose hardware, in a few years, even a software-only system might be adequate.

### IV.B. Build it and they will come

It is often assumed that if videoconferencing at low cost is possible, many conferences will be held (*build it and they will come*). We found that initially there was an enthusiastic audience for the conferences, but as the novelty wore off, the use of the system declined. In retrospect, this seems obvious. First, the participants in the conference all had other claims on their attention, and so were not available all the time. Without a ringing mechanism, it was easier to send electronic mail to the person than to call a videoconference (which, further, does not leave behind a written trail of thoughts). Second, in a geographically distributed network, time differences affect the ability of a person to participate in a conference. Even the three hour time difference between the East and West coasts of the US was a barrier to choosing a mutually agreeable time for the conferences. One can imagine the problems with a worldwide conference, where one or more parties may be liable drop off into deep sleep at any moment.

Thus, we believe that one should view videoconferencing not as a ‘‘killer app’’, but as a tool with a specific function. It is not likely to radically change the way in which we work because of the limitations pointed out above. We think that it is likely that after an initial surge in interest, users will not routinely use videoconferencing unless they use it to solve a particular problem. Further, without an improvement in the quality of the Internet tools, their mere presence is not going to result in the development of a large user base.

### IV.C. Video is more important than audio

It is felt that the presence of video in videoconferencing somehow elevates it above audioconferencing. However, in our experience, good quality audio was far more important than good quality video. The participants were willing to put up with poor video frame rates as long as the audio signal was steady. But, as soon as the audio quality degraded, the conference was called off. Thus, it appears that while the video component does help participants to sustain their interest, the audio component is essential. This has also been noted by others in the field [1]. The video component provides the look-and-feel for the videoconference, but it is the audio component that carries information.

## V. Suggestions for future work

In this section, we make recommendations for improvements, and point out some research topics in the general area of videoconferencing applications.

### V.A. Better signal processing for `vat`

In an earlier section, we described how `vat`’s sensitivity to background noise made it unusable without the ‘‘push to talk’’ option. One solution to the problem is to buy an expensive (˜1000 US dollars) directional microphone. A cheaper solution would be a better software filtering algorithm that eliminates noise consistently and correctly. This topic seems to be well understood by manufacturers of speakerphones, and the results obtained there could be applied.

### V.B. Conference control

There is a need for protocols that allow both for distributed moderation and aid a central moderator in conducting a conference. For example, if each participant had a display that showed that some other participant wanted to speak, then they could use this information in deciding who goes next. Coordination tools should also allow scheduling of conferences, and the ability to invite someone to join the conference. An example connection and resource management protocol is described in Reference [11]. Several problems arise in making the conference control protocol scale well with the number of users and level of interaction, which are described in Reference [10]

### V.C. Reduce resource requirements

In a well-run conference, only the current speaker needs to be seen, and only one or two people need to be heard. Thus, the resource requirement in the network is only to carry one or two audio streams, and a single video stream. In current technology, all the audio and video streams are carried to each receiver, who filters them. It would be more efficient if conference participants agreed to some ground rules so that their net resource requirement would be reduced. This idea of using ‘‘sharing groups’’ has been proposed by the Tenet group at Berkeley [4]. Since the Internet does not explicitly reserve resources, implementing sharing groups in the Internet would not need signalling support, only coordination between application programs. This would make it easier to implement sharing groups on the Internet then in a connection-oriented network.

Another way to reduce resource requirements

would be to implement a mixing service. This service, which would combine multiple audio signals into a composite signal, can reduce resource utilization, since only the mixed signal needs to be multicast. Mixing would also enable applications such as distributed music rehearsal, where each participant would play one track of the music, and all the tracks are mixed and multicast. A proposal for application level *combination nodes* is made in Reference [10].

### V.D. Improved user interface

As noted above, the current user interface leaves much to be desired. Making the interface more intuitive would make it more acceptable to lay users. We believe that it should be as easy to use videoconferencing tools as it is to use a telephone.

### V.E. Adaptive flow control

The bandwidth available to an endpoint over the Internet is a dynamically varying quantity. However, `nv` and `vat` ignore this and send as fast as they can capture and compress information (subject to a user-specified upper limit for `nv`). Instead, they could monitor the state of the network, and modify compression parameters (such as the quantization level) to adapt their bandwidth requirement to available capacity. This would allow users to get better performance on a lightly loaded network, and have a graceful degradation in performance as the network load increased. Techniques to adapt the compression factor and to adapt the endpoint load to the network state have been described in recent work [8].

## VI. Summary

We have used some easily available Internet tools, `nv`, `vat` and `sd` over a high speed ATM wide area network for large videoconferences. Our experience revealed several problems with these tools, and we have found that some common intuitions about videoconferencing are only myths. We also offer some suggestions to improve the performance of these tools and areas for research.

## VII. Acknowledgements

We would like to thank John Lockwood for setting up the MBONE over Xunet II, installing the Internet tools, and patiently explaining to us how to use them. He is single-handedly responsible for our videoconferencing facilities. Thanks also to Van Jacobson, Steve McCanne and Ron Fredericks for providing the Internet community with the tools described in this paper. Dave Kristol and Henning Schulzrinne provided useful comments on an earlier draft. Bill Marshall and Pat Parseghian kept Xunet operational despite our best efforts to crash it.

## VIII. References

1. J. S. Angiolillo, H. E. Blanchard and E. W. Israelski, Video Telephony, *AT&T Technical Journal*, May/June 1993.

2. A. Berenbaum, M. J. Dixon, A. Iyengar and S. Keshav, Design and Implementation of a Flexible ATM Host Interface for XUNET II, *IEEE Computer Network Magazine*, July 1993.

3. R. Fredericks, `nv-3.2` network video tool, *Available for anonymous FTP from ftp.parc.xerox.com*, 1993.

4. A. Gupta and M. Moran., Channel Groups: A Unifying Abstraction for Specifying Inter-stream Relationships, Tech. Rpt.-93-015 , International Computer Science Institute, Berkeley, CA , March 1993. .

5. V. Jacobson and S. McCanne, `vat-2.17` audioconferencing tool, *Available for anonymous FTP from ftp.ee.lbl.gov*, 1993.

6. V. Jacobson and S. McCanne, `sd` session directory tool, *Available for anonymous FTP from ftp.ee.lbl.gov*, 1993.

7. V. Jacobson and S. McCanne, `wb` distributed whiteboard tool, *Available for anonymous FTP from ftp.ee.lbl.gov*, 1993.

8. H. Kanakia, P. P. Mishra and A. Reibman, An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport, *Proc. ACM SigComm*, 1993.

9. S. McCanne, `vic` network video tool, *Alpha version not available for general release* , 1993.

10. E. M. Schooler, The Impact of Scaling on a Multimedia Connection Architecture, *Multimedia Systems 1*, 1 (1993), 2-9.

11. E. M. Schooler, Case Study: Multimedia Conference Control in a Packet-switched Teleconferencing System, *Journal of Internetworking Research and Experience 4* (June 1993), 99-120.

## Author Information

Srinivasan Keshav is with the Computer Science Research Center at AT&T Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974 USA. He received a Bachelor's degree from the Indian Institute of Technology, Delhi in 1986, and a PhD from the University of California at Berkeley in 1991, both in Computer Science. His research interests are in flow and congestion control, quality of service in broadband networks and native-mode ATM protocol stacks.