Conferencing and Collaborative Computing

Henning Schulzrinne
GMD Fokus, Berlin
hgs@fokus.gmd.de

August 10, 1995

Abstract

Despite their long research history, few synchronous conferencing tools have found widespread deployment and use. In the area of application sharing, the diversity of platforms and issues of security and state maintenance are predominant. The paper discusses some of the areas where further research and engineering work is necessary, particularly for the audio and video components of multimedia conferences. Hardware support needs to be designed to avoid unnecessary high-rate data transfer through system bus or memory. Operating systems need to be aware of the scheduling requirements of continuous-media applications, which differ from both those of classical computing applications and industrial real-time systems. Networks need to be designed to provide end-to-end quality of service assurance and scalable multicast. Since the diversity of conferencing applications far exceeds that of small-group sessions, mechanisms to reuse conferencing applications in a variety of settings have to be developed.

1 Introduction

This paper attempts to provide a personal perspective on some of the problems, issues and possible directions for conferencing and collaborative computing. As with all such endeavors, the result is likely to be biased by the personal experiences and prejudices of the author. References are mostly pointers to recent related work, where more comprehensive discussions may be found.

Given the long history of efforts in both conferencing (from the telephony and computer side) and computer-supported cooperative work, the number of tools actually in regular and commercial use is not that large:

- electronic mail
- bulletin boards (Usenet news, Lotus Notes)
- screen sharing tools (e.g., ShowMe from SunSoft)
- text-based conferencing systems like Internet Relay Chat, multi-user dungeons (MUDs) or similar systems offered by commercial information providers like CompuServe or America Online

- telephone conferences
- conference room and roll-about video conferencing systems using multipoing control units (MCUs or bridges) for usually no more than three or four participants

In the last few years, it appears that almost every self-respecting computer science department and government-funded project has developed a multimedia-conferencing tool, particularly now that many workstations and personal computers and even some X terminals are capable of audio I/O and at least slow-scan video output. However, this increased activity has still tended to concentrate on small-scale videoconferencing, suitable for no more than a dozen or so participants. Sometimes, it appears that the functionality of these conferencing systems is guided by the limited experiences of the developers, consisting of seminars and developer's meetings taking place in dedicated meeting rooms. Video conferences meant two to four talking heads in rectangles on a screen.

However, recently, there have been attempts to support the richness of human communication situations, such as unplanned hallway encounters, drop-in seminars, panel discussions, jury trials, lectures, pay-TV, and more. However, just mapping familiar communication patterns into conferencing systems may not be enough. The challenge is to progress beyond the "horse-drawn carriage" stage of system building, making use of the possibilities of workstation-based conferencing that cannot be mapped into physical conferencing situations.

2 Audio and video

Video conferencing is often portrayed as the "killer application" that is going to motivate everyone to go out and purchase whichever hot new networking or workstation technology is being proposed. However, both practical experience and formal experiments [1] seem to show that the use of video in conferencing applications is somewhat secondary. Video does not affect task performance; at least in technical discussions, it is often used to display viewgraphs (rather poorly) and to indicate how many people are still physically present within the conference. These tasks can also be filled by video with a few frames per second. For conferences with more than three or four participants, screen real-estate quickly runs out, particularly if other applications such as shared editors or drawing spaces are to be used. The ability to quickly resize individual images either manually or automatically, e.g., based on audio or drawing activity, can help. It may mean that video conferencing rooms with multiple, large high-resolution screens may continue to be necessary even after every workstation is video-equipped. The widespread use of video cameras at home seems to have made their presence less intimidating. On the other hand, good quality audio, with true full duplex communication and echo cancellation, possibly enhanced with spatial cues [2–4], appears to have been neglected. Good quality audio and video will prove particularly challenging in workstation-based conferencing since offices are often noisy, have acoustically reflective walls and lighting is inappropriate.

3 Hardware Support

Currently, real-time multimedia services are provided both by general-purpose processors, without assistance beyond analog/digital conversion, and special-purpose add-in boards. It remains to be seen whether support for video and high-quality audio is closer to network services, where protocol processors have fallen out of favor, or graphics engines, where they have become standard in even low-end PCs. The expansion of video and high-quality audio data from encodings like JPEG or MPEG to pixel data argues for keeping it off any shared buses, and probably out of the memory system. (General RAM speeds have not increased significantly beyond the 70 ns mark.) Also, unlike network packets, the result of video decoding is usually not processed further, beyond moving it on the screen or simple scaling. On the other hand, it is far easier to introduce a new video coding if it can be distributed as software. For encoding, particularly motion estimation, only hardware-assisted solutions with low-level parallelism appear to be feasible. Work in developing video codecs suited for software encoding and decoding continues [5], but high coding gains seem to require motion estimation. Interoperation between dedicated circuit-switched codecs and workstation-based video conferencing is feasible with software-implementations of video coding standards like H.261. Desirable are also codecs that work reasonably well for both "natural" data like human faces and clothing as well as computer-generated material such as text on slides.

4 Operating System Support

Neither standard multitasking nor real-time operating systems are well matched to the needs of real-time multimedia conferencing. Standard Unix scheduling favors I/O intensive processes, while software-based codecs may not get sufficient computational resources. Real-time operating systems provide the desirable means to lock processes into memory, but their rigid scheduling disciplines demand a detailed knowledge of the CPU requirements of the applications, something which usually neither application programmer nor user has. To make matters worse, the computational requirements often depend on a complex interaction of system resources (e.g., 8-bit vs. 24-bit screen for dithering, or system cache speed), user settings (frame rate and image size), encoding, image size and even program material. It has been proposed to make applications learn about their resource needs [6], but the success of that appears doubtful, given that for a video application, for example, CPU usage depends, among other factors, on the frame rates and number of sending sources, which may be changing constantly. Attempts to modify existing operating systems by adding a real-time scheduling class have caused system instabilities and had to be abandoned [7]. While ideally many real-time sources are periodic, network delays may cause bunching of packets. If CPU resources are made available only periodically, only parts of these packet bursts will be processed, leading to unacceptable quality degradation. Other real-time sources like slow-scan video may not be periodic at all. Overall, there remains work to be done to arrive at robust scheduling policies that guarantee sufficient resources to real-time applications without starving others and without having to have precise knowledge of application demands. Real-time applications may require kernel services, so that their scheduling has to be treated carefully. It may be necessary to delegate compute-intensive real-time tasks to dedicated processors with low utilization.

While POSIX and X11 allow reasonably quick cross-platform development on Unix systems for many interactive applications, every vendor seems to have their own audio and video programming interface. Some attempts have been made at cross-platform APIs (NCD netaudio, DEC audiofile), but these appear to have a number of shortcomings. Most audio APIs seem designed mainly for playing back audio clips rather than real-time use. Netaudio and audiofile also abandon the Unix device-as-file model, requiring separate hooks into event handlers, separate read/write routines, and so on. This appears to be a step backwards and makes it difficult to use the same code to read from an interactive audio device and a file containing audio data.

An audio API should allow sharing of the single speaker by several applications by either mixing at different volumes or priority override. External applications like VU meters or automatic gain control should be attachable to the audio input and output, without having to be replicated for every application. Separating audio source and sink by a network requires sophisticated playout adjustment at the receiver, particularly if the solution is to be usable beyond an uncongested local area network. Thus, a general system solution seems preferable to every application having to develop their own solution. Difficulties remain; for example, indicating the current talker is more complicated, as is the compensation for losses or other interventions by the application. Thus, either the audio library has to perform almost all desirable audio services or very few beyond mixing, volume control and the like.

Also, for video, a clean separation of video frame grabbing and encodings appears difficult if extraneous normalizations and copies are to be avoided.

5 Network Support and Protocols

Deployed networks are ill-suited for truly flexible computer conferencing. Synchronous networks like ISDN, switched 56 kb/s or higher-speed lines and the protocols for these networks like H.320 suffer from the difficulty of allocating bandwidth dynamically among a time-varying mix of applications. Multicast requires multipoint control units (MCUs) and is limited to fairly small group sizes. Framing protocols like H.221 are difficult to handle in software. Deployed packet networks like the Internet offer much more flexibility, but poor quality if even a small fraction of the overall bandwidth is used by real-time services. (The German X.25 network underlying the German research Internet seems particularly unsuited for real-time services, even though reliable transport protocols like TCP show acceptable performance.) Strategically placed users can, intentionally or not, flood the network with video traffic. More robust forms of multicast (rather than the currently-deployed truncated broadcast) need to be developed and deployed, possibly with different protocols for sparse and dense groups. The interoperation of receiver-oriented IP multicast with sender-rooted multicast found in ATM remains unsolved for anything but small, local domains. Diagnostic tools are just starting to appear.

ST-II [8] has been proposed as an alternative solution, but it suffers from implementation complexity and poor scaling since the sender has to explicitly establish a connection to every end point. Using ST-II over ATM, both being connection-oriented, would appear to be a promising approach, but the two intertwined connection establishment phases add further delay and complication [9].

From the beginning, ATM was billed as a true multimedia service, with guaranteed quality of service. So far, commercially available switches offer at most a simple priority mechanism and thus cannot offer guarantees unless the combined peak rates of all VCs is less than capacity. (Clearly, any network technology can offer good service under those conditions.)

The debate over the proper role of ATM continues, with approaches ranging from treating ATM like adjustable-rate point-to-point links in a connectionless internetwork (classical IP over ATM [10]) to connectionless servers, i.e., yet another network-layer protocol, to dynamic connection establishment similar to the approaches used for routing IP over X.25 and ISDN, and finally pure ATM end-to-end solutions, possibly directly carrying MPEG streams, without any network layer protocol. Initial multicast support of switches and signaling protocols appears to be poor, with at best only homogeneous 1-to-N multicast in the offing. (Also, while theoretically supported, parallel connection establishment seems to be rather limited in scope, leading to extremely long connection-establishment times.) Receiver-initiated signaling as proposed for IP (RSVP, [11]), ST-II and ATM could relieve hosts from the burden of managing dynamically changing multicast groups of several hundred end systems. RSVP, in particular, offers an interesting approach to establishing state in datagram networks, but its error recovery properties and interaction with routing need to be explored.

At the transport level, all-in-one combined network/transport protocols like various derivatives of XTP [12] compete with simple special-purpose protocols like RTP [13] for the unreliable delivery of real-time data. The complexity, large header size and lack of routing support for protocols like XTP (without being encapsulated) have limited their use to demonstrations. RTP, based on packet-voice work dating back into the 1970s and more recent extensions to IP multicast by Jacobson, has also attempted to accommodate parties beyond the actual media agents, like quality-of-service monitors, recorders, firewalls, filters, bridges and the like.

Synchronization between media streams has been the topic of a large body of research [14–16]. It appears, however, that once synchronized clocks are available with clock differences of a few milliseconds [17], the problem is largely solved. Explicit synchronization algorithms appear to provide special-case multi-party clock synchronization. Some further experimental work on playout synchronization in different networks is useful, although the need for sophisticated algorithms arises primarily in lower speed packet networks like the Internet [18].

The integration of resource discovery tools like the World-Wide Web with multimedia conferencing and collaborative computing is just beginning to be explored.

6 Structure

Beyond providing for individual media sessions, a number of researchers have explored metaphors for conferences to provide structure and navigation. Examples are virtual meeting rooms [19, 20] and hierarchical conferences, where a conference can itself be a member of a conference [21]. Room-based conferences can readily be represented by the mapping of conferences onto WWW pages [22], with the additional ability to use proximity on the page to indicate full or partial communication. (Participants normally communicate only with those they “touch” on the WWW page, but can “shout” to anyone on the page.) Also, operations on conferences as objects, such as adding or merging, [21] have been studied. It remains to be seen whether these models are flexible enough to encompass most real-life communication situations and whether the additional complexity is warranted and can be represented to the user in meaningful terms. The wholesale moving of participants as implied by conferences as objects may run counter to the desire of individual control on the part of participants.

7 Conference Control

The scope of the term conference control is not defined precisely. It is generically applied to those aspects of computer-mediated communication, particularly synchronous ones, that are not concerned with data transport but rather with providing a structured, dynamic framework for a number of media sessions and a set of users and auxiliary tools. Conference control establishes agreement on common state (e.g., set of permissible audio encodings, the identity of a moderator or access rights (floor control)), helps with adding new users to a session and reserves necessary network resources. Conference control may be layered, in that simple, per-media stream session management is utilized by a higher-layer protocol to tie together several stream. Clearly, conference control is related to signaling in the telephony world, particularly the notion of calls consisting of bearer services and aspects of negotiation.

Many conferencing systems (like other CSCW systems) had a rather idealized notion of how the "real world" works: a registration, admission and negotiation phase, followed by the conference proper, finally the closing. Even more so than in physical conferencing, however, it appears that electronic, particularly workstation-based conferences are far more fluid, with participants joining and dropping out of individual media sessions or the whole conference, taking phone calls, answering questions from people walking into their offices, etc.

Given a large number of participants and the complexity of applications and networks, it is likely that parts of a conference will misfunction during a session. Rather than terminating the conference, it will usually be more desirable to continue with as many participants as possible. The "sticky" conference control protocol [23] puts particular emphasis on such robustness. It appears that in many cases it is unrealistic to expect to have a universally agreed-upon state. Internet multimedia conferencing, exemplified by applications like vat, nv, NeVoT [24] and wb, take this to the extreme of simply periodically announcing the presence and state of participants via multicast. The announcement period is varied randomly to avoid synchronization, with increasing mean as the number of participants increases. For an audio conference with a media packet rate of 50 packets per second, about one hundred sites can be supported at an overhead of 1% and an awareness-lag of about one minute. For very large conferences of thousand or more participants, "selective awareness", indicating recent or signed-up speakers, may be needed, if for no other reason that displaying a list of a thousand participants on a screen is not very useful.

In this model, conferences consisting of several media sessions are announced through a multicast session directory (for example a tool called `sd`) by the initiator. This style of conference also takes the approach of controlling the delivery of data at the receiver rather than through, say, remote control. Similarly, access is limited by encryption and key distribution rather than an invitation process often found in small-scale conferences. This follows the basic philosophy of open networks that only encrypted data is truly safe from intruders.

Given the wide diversity of collaboration styles, it appears difficult to arrive at a canonical, all-encompassing conference control protocol or application. Rather more promising appear attempts to distill common, underlying control functions and allow combining these in appropriate ways. The agreement protocols [25, 26] formalizes the problem of agreeing on shared state and modifying it based on voting rules. If eventual consistency is desired, some form of reliable multicast is necessary, an area where many protocols have been proposed [**?**, 28, 29]. For distributing state, reliable multicast has to deal gracefully with sites joining and leaving. It would be undesirable to halt a shared editor, say, if one of the participants is disconnected without formal connection termination.

On the other hand, if two disconnected parts of a group continue to modify shared state independently, it may be impossible to arrive at a reasonable single shared state after the disconnection heals.

State agreement covers the functionality of floor and access control, media negotiation, moderator selection, and similar tasks. Other functional areas that need to be integrated into an overall architecture include directory services for sessions and invitation services. Other generic network services, such as encryption key distribution or user location services, are particularly valuable for computer-mediated communication.

8 Tools and Applications

Collaborative computing will use both tools specially designed for use within conferences and generic tools. The latter is generally desirable as conferences are not the most opportune time to experiment with unfamiliar applications. Generic mechanisms for application sharing have been developed for X11 [30].

Even if special multi-user tools are used, the same tools should be usable for both small and large conferences. Having to switch tools because a small discussion turned into a seminar is rather undesirable. Unfortunately, this may be difficult to realize, as techniques like a fully meshed net of reliable transport connections do not scale, but are easier to program than reliable multicast. Similarly, tools should work both in high-bandwidth, low-loss, low-delay local area networks as in wide-area networks with the opposite properties, even if with reduced functionality or quality. This does impose a burden of having to compensate for much poorer network service. For some tools such as shared editors, integration of synchronous and asynchronous operation into a single tool is helpful. For others, it may be preferable to have "attachable" applications such as recorders and playback devices.

Sharing application at a relatively low level like a windowing system imposes the same user interface on all participants, regardless of personal preferences and local capabilities. Allowing for different user interfaces implies the rather difficult task of defining abstract operations and common data formats or state descriptions.

While traditional telephony can guarantee access limitations and adequate privacy by appropriate connection setup, many packet networks have no effective way from keeping users from sending or even listening to a particular packet stream. Thus, the only effective means of limiting distribution is to give the receiver control over what is rendered on the local workstation. Privacy has to be ensured by encryption. This receiver orientation is made easier if mixing, for example is performed at the end system rather than at a multipoint control unit. Similarly, coupling of several media, e.g., having the video display follow audio activity, is best accomplished at the end system since it can implement any desired policy. (Filtering unwanted data in the network, however, maybe more difficult.) IP multicast is naturally receiver-oriented, but connection-oriented protocols like ST-II or ATM still require signaling from the receiver to the sender, with appreciable delays. (See discussion of RSVP above.) Despite the basic notion of receiver control, mechanisms for voluntary remote control of applications, including senders, can be helpful.

In terms of applications, we are just gaining first experience with conferences (actually, mostly seminars, lectures and the like) in the Internet scaling from two to several hundred. Beyond quality problems due to networks without resource reservation, operational problems for more complicated conference scenarios remain to be addressed. Invitation and coordination of speakers are among the issues.

Experience suggests that for application writers, receiver-oriented multicast is much easier to implement than explicit participant lists. Also, the same is true for retransmitting state periodically rather than explicitly exchanging state with new arrivals, albeit at the cost of larger delays.

Given the diversity of conference control mechanism and the wide range of applications, a flexible means of tying together media applications, supporting tools (recorders and playback applications, calendars, etc.) and control applications (floor control, conference control, etc.) needs to be found. It may be nice if applications can be locally distributed among hosts, but still appear as belonging to a single user and be directed by a single conference controller. In Unix-like systems, either linking libraries or communicating processes may be used, but only the latter is usually configurable by the user. Traditional interprocess communication is not suitable since in many cases, there is no clear 'client' and 'server' relationship [31,32]. Also, the same kind of information, say, about new sessions or participants, may be of interest to a number of applications. This suggests multicast. However, local IP multicast requires a clear division of messages into classes by the sender, something which appears to be difficult for conference control. If more than one multicast address is used, a directory and address allocation service is needed. As an alternative, the Network Voice Terminal (NeVoT) is exploring the use of 'application-level' multicast, where a messages are forwarded to a central replicator called pmm (pattern-matching multicastor) containing regular-expression filters installed by applications. Applications connect to the replicator, which forwards messages to interested parties. Messages describe operations on objects such as conferences, media sessions, and the like. This approach allows to built conferencing applications without source modification or to attach new tools to existing conferencing systems [33].

9 Conclusion

Multimedia conferencing has long been considered a prime application for high-performance workstations and networks. However, actual use has consistently fallen short of expectations. Rather than fundamental technical advances, a number of coordinated engineering improvements in the areas of workstation architecture, operating systems, programming support, conference control and networks are necessary.

References

[1] S. Gale, "Adding audio and video to an office environment," in *Studies in computer supported cooperative work* (J. M. Bowers and S. D. Benford, eds.), Human Factors in Information Technology, pp. 49–62, Amsterdam: North-Holland, 1991.

[2] T. Becker, "Konzeption und realisierung eines "virtuellen konferenztisches"," Master's thesis, Technische Universit t Berlin, Berlin, Germany, Dec. 1993. Studienarbeit.

[3] S. Hayashi, "Increase in binaural articulation score by simulated localization using head-related transfer function," *IEICE Transactions on Fundamentals*, vol. E75-A, pp. 149–154, Feb. 1992.

[4] S. Masaki, T. Arikawa, H. Ichihara, M. Tanbara, and K. Shimamura, "A promising groupware system for broadband ISDN: PMTC," *ACM Computer Communication Review*, vol. 22, pp. 55–56, Mar. 1992.

[5] R. Frederick, "Experiences with real-time software video compression," in *Sixth International Workshop on Packet Video*, July 1994.

[6] M. B. Jones, "Adaptive real-time resource management supporting modular composition of digital multimedia services," in *Proceedings of the 4th International Workshop on Network and Operating System Support for Digital Audio and Video*, (Lancaster, U.K.), pp. 11–18, Lancaster University, Nov. 1993.

[7] J. Nieh, J. G. Hanko, J. D. Northcutt, and G. A. Wall, "SVR4 UNIX scheduler unacceptable for multimedia applications," in *Proceedings of the 4th International Workshop on Network and Operating System Support for Digital Audio and Video*, (Lancaster, U.K.), pp. 35–47, Lancaster University, Nov. 1993.

[8] C. Topolcic, "Experimental internet stream protocol, version 2 (ST-II)," Request for Comments (Experimental) RFC 1190, Internet Engineering Task Force, Oct. 1990.

[9] O. Hagsand and S. Pink, "ATM as a link in an ST-2 internet," in *Proceedings of the 4th International Workshop on Network and Operating System Support for Digital Audio and Video*, (Lancaster, U.K.), pp. 189–198, Lancaster University, Nov. 1993.

[10] M. Laubach, "Classical IP and ARP over ATM," Request for Comments (Proposed Standard) RFC 1577, Internet Engineering Task Force, Jan. 1994.

[11] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "Rsvp: a new resource ReSerVation protocol," *IEEE Network*, vol. 7, pp. 8–18, Sept. 1993.

[12] B. Metzler and I. Miloucheva, "Specification of the broadband transport protocol XTPX." R2060/TUB/CIO/DS/P/001/b2, Feb. 1993.

[13] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications." Internet draft (work-in-progress) *draft-ietf-avt-rtp-*.txt*, Mar. 1995.

[14] S. Ramanathan and V. P. Rangan, "Adaptive feedback techniques for synchronized multimedia retrieval over integrated networks," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 246–260, Apr. 1993.

[15] D. C. A. Bulterman, "Synchronization of multi-sourced multimedia data for heterogeneous target systems," in *Third International Workshop on network and operating system support for digital audio and video*, (San Diego, California), pp. 110–120, IEEE Communications Society, Nov. 1992.

[16] T. D. C. Little, A. Ghafoor, C. Y. R. Chen, C. S. Chang, and P. B. Berra, "Multimedia synchronization," *The Quarterly Bulletin of the IEEE Computer Society Technical Committe on Data Engineering*, vol. 14, pp. 26–35, Sept. 1991.

[17] D. L. Mills, "Internet time synchronization: the network time protocol," *IEEE Transactions on Communications*, vol. 39, pp. 1482–1493, Oct. 1991.

[18] R. Ramjee, J. Kurose, D. Towsley, and H. Schulzrinne, "Adaptive playout mechanisms for packetized audio applications in wide-area networks," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, (Toronto, Canada), June 1994.

[19] S. R. Ahuja and J. R. Ensor, "Call and connection management: making desktop conferencing systems a real service," *ACM Computer Communication Review*, vol. 22, pp. 10–11, Mar. 1992.

[20] T. O'Grady and S. Greenberg, "A groupware environment for complete meetings," in *Proceedings CHI'94*, pp. –, 1994.

[21] H. M. Vin and P. V. Rangan, "System support for computer mediated multimedia collaborations," in *Proceedings of the 1992 ACM Conference on Computer Supported Cooperative Work (CSCW'92)*, (Toronto, Canada), pp. 203–209, ACM, Nov. 1992.

[22] E. Shapiro, "Virtual places - a foundation for human interaction," in *Proc. of the Second World Wide Web Conference'94*, (Chicago, Illinois), Oct. 1994.

[23] C. Elliott, "A 'sticky' conference control protocol," *Internetworking: Research and Experience*, vol. 5, pp. –, 1994.

[24] H. Schulzrinne, "Voice communication across the Internet: A network voice terminal," Technical Report TR 92-50, Dept. of Computer Science, University of Massachusetts, Amherst, Massachusetts, July 1992.

[25] S. Shenker and A. Weinrib, "Managing shared ephemeral teleconferencing state: policy and mechanism." memorandum, Mar. 1994.

[26] B. Rajagopalan, "Consensus and control in wide-area group communication." unpublished memorandum, Nov. 1993.

[27] M. Handley and I. Wakeman, "CCCP: conference control channel protocol – a scalable base for building conference control applications." V1.4, Mar. 1994.

[28] L. L. Peterson, N. C. Bucholz, and R. D. Schlicting, "Preserving and using context information in interprocess communication," *ACM Transactions on Computer Systems*, vol. 7, pp. 217–246, Aug. 1989.

[29] R. Aiello, E. Pagani, and G. P. Rossi, "Causal ordering in reliable group communications," in *SIGCOMM Symposium on Communications Architectures and Protocols* (D. P. Sidhu, ed.), (San Francisco, California), pp. 106–115, ACM, Sept. 1993. also in *Computer Communication Review* 23 (4), Oct. 1992.

[30] H. Abdel-Wahab and K. Jeffay, "Issues, problems and solutions in sharing X clients on multiple displays," *Internetworking: Research and Experience*, vol. 5, pp. 1–15, Jan. 1994.

[31] J. Crowcroft, “Remote procedure call: not a panacea for distributed computing problems.” University College London, Feb. 1993.

[32] M. Roseman and S. Greenberg, “Building flexible groupware through open protocols,” in *Proceedings COSC’93*, pp. –, ACM, 1993.

[33] H. Schulzrinne, “Dynamic configuration of conferencing applications using pattern-matching multicast,” in *Proc. International Workshop on Network and Operating System Support for for Digital Audio and Video (NOSSDAV)*, (Durham, New Hampshire), pp. 231–242, Apr. 1995.