The Caelum[*]Toolkit for CSCW: The Sky is the Limit[†]

Tal Anker   Gregory V. Chockler   Danny Dolev   Idit Keidar

{anker,grishac,dolev,idish}@cs.huji.ac.il
http://www.cs.huji.ac.il/{~anker,~grishac,~dolev,~idish}
Institute of Computer Science
The Hebrew University of Jerusalem
Jerusalem, Israel

Abstract

We present a general framework for the construction of groupware and computer supported cooperative work (CSCW) *applications. Examples of such applications include: multi-media and desktop conferencing, distance learning, interactive games and simulations, and collaborative computing. We provide application builders with a* software development kit (SDK) *that supports sharing of a variety of applications among dynamically changing groups of users. We provide a variety of efficient communication solutions, tuned towards different* quality of service (QoS) *requirements, as well as tools for maintaining consistency of distributed and replicated information in the face of faults. We exploit the group communication paradigm for dynamic discussion groups, and for keeping track of the dynamically changing set of participants. The discussion groups may be organized hierarchically, and discussion in groups may be secure. Our services are fault tolerant and scalable, and are therefore appropriate for multiprocessor failure prone networks such as the Internet.*

1 Introduction

Reliability carries different meanings for different applications. For example, in a replicated database setting, reliability means that messages arrive in the same order at all sites, and are never lost. In order to guarantee this reliability property, it is acceptable to sacrifice real-time message delivery: some messages may be greatly delayed, and at certain periods message transmission may even be blocked. While this is perfectly acceptable behavior for a reliable database application, this behavior is intolerable for a video server. For video transmission, reliability means real-time message delivery, at a certain latency and bandwidth; It is acceptable for some messages to be lost, as long as the latency and available bandwidth comply with certain predetermined stochastic assumptions. Implementations of database style reliability (i.e. message recovery and order constraints) may violate these assumptions, rendering the video decoding algorithm (e.g., MPEG [21, 31, 7]) incorrect.

A desktop and multi-media conferencing tool [30], is a *Computer Supported Cooperative Work* (CSCW) application incorporating various activities such as video transmission and management of replicated work space. These activities obviously require different qualities of service, and yet are part of the same application. CSCW applications also require directory services as well as security and billing services. Furthermore, it is desirable for a CSCW application in a multi-processor network to be fault-tolerant, and to support smooth reconfiguration when parties join or leave. Figure 1 depicts a video conferencing application, with a new party wishing to join the discussion.

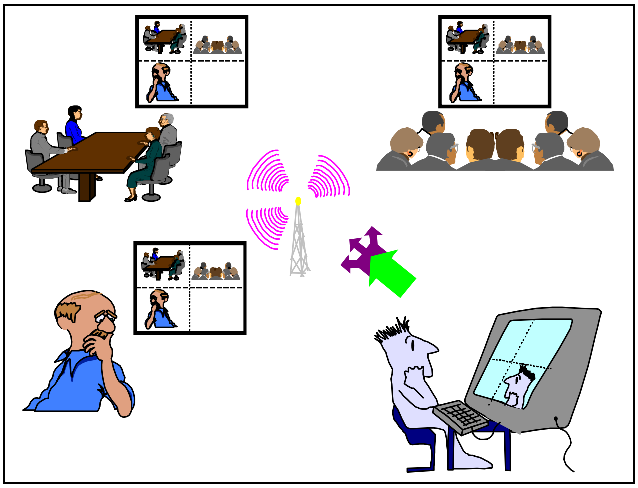

Figure 1: A New Party Joining a Video Conference

In this paper we suggest a comprehensive tool,

[*]Caelum is the Latin for sky.

[†]This work was supported by ARPA grant number 030-7310 and by the Israeli Ministry of Science grant number 032-7658.

Caelum, for the development of highly available groupware and CSCW applications. Caelum is geared towards multi-process failure prone environments (e.g., the Internet). The suggested solution integrates all the aspects of the development of such applications: it provides efficient communication solutions for a variety of *quality of service (QoS)* requirements, as well as tools for maintaining consistency of distributed and replicated information in the face of faults. It supports multi-party conferencing in dynamic discussion groups, while keeping track of the dynamically changing set of participants. The groups may be organized in a hierarchical directory.

Caelum allows the application to define various input filtering policies. For example, in an interactive classroom, only the teacher is allowed to provide voice input and to draw on the board. The teacher is able to temporarily delegate this authority to one of the students. On the other hand, in a text chat application, input concurrently arrives from multiple users. In addition, Caelum provides a variety of useful utilities and services, e.g., security and billing services.

1.1 The Caelum Architecture

Caelum provides application builders with a comprehensive *software development kit (SDK)* that supports sharing of a variety of applications among dynamically changing groups of users. Caelum is designed to operate in a heterogeneous multi-processor environment. Caelum provides the tools for overcoming machine and network failures.

Caelum exploits a novel concept: a *Multi-media Multicast Transport Service (MMTS)* [12], that supports multiple QoS group communication options[1]. This makes the services of Caelum inherently fault tolerant, and allows the application builder to define the tradeoffs between the level of synchronization/reliability and the timeliness of message delivery.

The Caelum architecture consists of a toolkit front-end, a membership service module, an MMTS module, and a module providing session-level services. The architecture of Caelum is described in Figure 2.

We are currently implementing Caelum using the Transis [14] and Ensemble [20] group communication systems, enhanced with QoS communication. Some components of Caelum were previously implemented in C, and our current effort focuses on implementing parts of Caelum in OCAML, and providing user interfaces in C, Java and OCAML.

[1]In [12] the membership services are regarded as part of the MMTS. Here, we follow the approach taken by Maestro [9], which separates the group multicast services from the membership services.

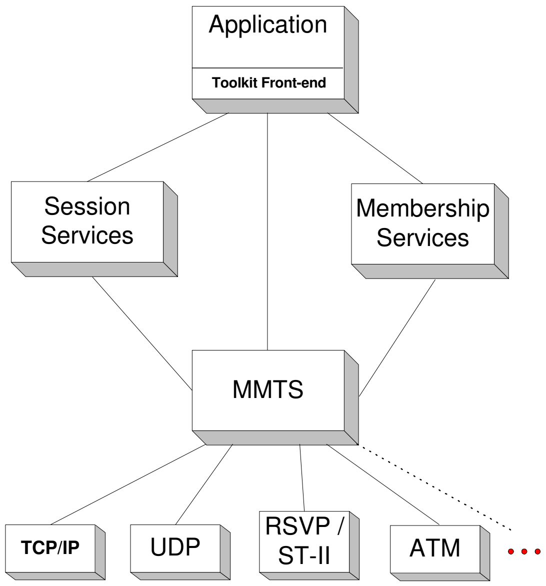

Figure 2: The Caelum Architecture

In Section 2 we describe the group communication paradigm, and in Section 3 we describe the MMTS. The membership services of Caelum are described in Section 4, and the session services in Section 5. In Section 6 we describe several examples of groupware applications, and how they may benefit from the services of Caelum.

2 The Group Communication Paradigm

Group communication is a powerful paradigm for the development of fault-tolerant distributed applications and for CSCW groupware and multi-media applications. It introduces the notion of *group abstraction* that allows processes to be easily arranged into multicast groups. A multicast group is identified by the *logical name* assigned to it when the group is created. Each message targeted to the group's logical name is guaranteed to be delivered to all the currently connected and operational group's members. This allows to handle a set of processes as a single logical connection. Furthermore, processes may dynamically join or leave these groups.

Group communication systems (GCSs), (e.g., Transis, Horus and Totem [1], Ensemble [20], ISIS [11], RMP [38], Phoenix [24] and Newtop [16]) provide group multicast and *membership* services. The membership service provides the group members with an indication of the current *membership*: the group of currently connected processes.

GCSs provide the application builder with various types of efficient reliable multicast, e.g., the *causal* multicast service guarantees that the reply to a message is never delivered before the message itself at any target. The *totally ordered* multicast service extends the *causal* service in such a way that all messages are delivered in the same order at all their targets.

GCSs today have begun to exploit new technologies, and to run over fast networks e.g., ATM and WAN environments. Unfortunately, the reliability introduced by the GCS (message recovery and ordering) requires message buffering and flow control mechanisms which may violate the QoS properties provided by the underlying communication media. These properties are crucial for the performance of application that require *real-time* message delivery, e.g., video multicast.

3 Multimedia Multicast Transport Services (MMTS)

In [12] a novel concept is introduced: a *Multi-media Multicast Transport Service (MMTS)* that supports QoS group communication. The MMTS provides a framework for *synchronization* of messages with different QoS requirements. The Caelum toolkit exploits the MMTS concept.

The MMTS concept is particularly beneficial for applications that integrate services with a variety of QoS needs. For example, multi-media and desktop conferencing systems require *Quality of Service (QoS)* communication for video transmission. Nonetheless, such applications are concerned with more than just transmitting a stream of video: they need to exchange messages for connection establishment, dynamic group management, and *negotiation and re-negotiation* of *Quality of Service (QoS)* [28]. Furthermore, it is desirable to make such systems fault tolerant.

Recently, several emerging projects addressed the challenge of incorporating QoS communication into the framework of group communication. Maestro [9] extends the Ensemble [20] group communication system by coordinating several protocol stacks with different QoS guarantees. The Collaborative Computing Transport Layer (CCTL) [29] implements similar concepts, geared towards distributed collaborative multimedia applications. Both systems implement and elaborate the concept of MMTS.

The VIC [25] video conferencing tool over the MBone[2] is a flexible framework for packet video. This approach uses a *conference bus* for broadcasting the various media in a conference session (e.g., whiteboard media, audio, and video). In the VIC architecture, the conference bus may be used along with a *coordination tool*. The MMTS can be viewed as integrating both the conference bus and the coordination tool.

The MMTS concept is flexible, it can exploit various underlying communication protocols and technologies, e.g., RSVP [40], ST-II [36] and ATM QoS. Furthermore, it modularly supports integration of new QoS options, e.g., the cyclic UDP QoS [35] that was implemented as a protocol layer in the Horus system [37].

One of the important challenges that need to be addressed when using groupware toolkits for a multimedia application is how to combine services with strong semantics with the QoS required by the multimedia application.

The MMTS allows the user to provide optional *synchronization barriers* among streams of messages of different QoS types. Synchronization barriers are implemented using reliable messages. Using synchronization barriers, the user may enforce order semantics w.r.t. messages of different streams. These barriers may delay "faster" messages until the arrival of "slower" messages that they depend on. For soft real-time applications, that can tolerate some bounded delay, a best-effort synchronization mechanism is provided. The best-effort service delays the message delivery for some pre-defined timeout in order to try to synchronize the different channels used by the application. After this timeout, the message may be either discarded or delivered in spite of the lack of synchronization, according to the application's specification.

This concept generalizes the Δ-*Causal* communication mode [8, 39]. In this communication mode, messages may be lost. Each message has a *lifetime*, Δ, after which its data is no longer meaningful, hence the message may be discarded.

Another example of best-effort semantics is the cyclic UDP [35] *prioritized* best-effort message recovery mechanism. Cyclic UDP allows the user to specify priorities for messages. Messages with a higher priority are recovered before messages with a lower priority. Message recovery attempts are stopped after a certain timeout period. Cyclic UDP may be incorporated in the MMTS, (as described in [12]), by recovering lost messages only until the synchronization barrier message is delivered.

4 Advanced Group Membership Services and Policies

Caelum provides a hierarchical group membership service with support for group policies. These services are valuable building blocks for conferencing applications and interactive games.

[2]Information about the MBone can be found in http://www.best.com/ prince/techinfo/mbone.html.

The Caelum membership service may be implemented either as a separate process (daemon), as part of the Caelum library, or as part of the MMTS. Caelum is flexible enough to allow a variety of implementation options.

4.1 Group Membership Services

Caelum supports the group communication paradigm: it supports dynamic discussion groups, and provides information regarding the dynamically changing set of participants. Each message targeted to the group's logical name is guaranteed to be delivered to all the currently connected and operational group's members.

The Maestro [9] system provides a common membership service for groups of processes running Ensemble [20] with different QoS options. The design of the membership service of Caelum was inspired by Maestro.

The basic membership service of Caelum is based on the CONGRESS CONnnection-oriented Group-address RESolution Service [6], which is designed for ATM networks, but may be exploited in other networks as well. CONGRESS supports two types of services: *address resolution*, which is a single query about the group membership, and *incremental updates*, which provides the user with updates every time the group membership changes.

CONGRESS provides basic efficient group resolution services for performance driven applications. It does not impose agreement on the order of membership changes, thus different members may incur the same membership changes in different orders. Furthermore, CONGRESS does not deal with message transmission, and hence does not impose any semantics on message ordering w.r.t. to membership changes.

Caelum allows applications that require consistency to agree upon the order of membership changes, and thus incur membership changes in the same order. This is done by using CONGRESS incremental updates in conjunction with a one round agreement protocol. Such order properties are provided by many group membership systems and algorithms (e.g., Cristian and Schmuck's one round membership protocol [13]). The agreement protocol is run only for groups that explicitly request this service.

Group communication systems usually provide strong semantics of message ordering w.r.t. membership changes, e.g., virtual synchrony [10, 17, 27]. Virtual synchrony requires synchronization among the applications and the membership service. This service is costly: it incurs a delay period in which messages may not be transmitted [17]. This synchronization greatly facilitates the design of applications that require consistency (e.g., applications with shared data [10, 3, 23, 5]), but is too costly for applications that require real-time message delivery (e.g., video transmission).

Caelum provides virtually synchronous communication for groups and message types that explicitly request this service. The implementation of Virtual Synchrony is based on synchronization messages, as depicted in Figure 3. The synchronization messages are represented by dashed lines, and the membership message is represented by a solid line. Before installing a new membership, the membership module sends a BLOCK message to the application. The application responds with a FLUSH message sent via the MMTS. After the FLUSH message, the application stops sending messages with a virtually synchronous QoS. The MMTS delivers the FLUSH message to the membership module after all the messages that were sent before it. Once the membership module receives the flush message from all the members of the new membership, it delivers the new membership, and the application may resume sending messages.

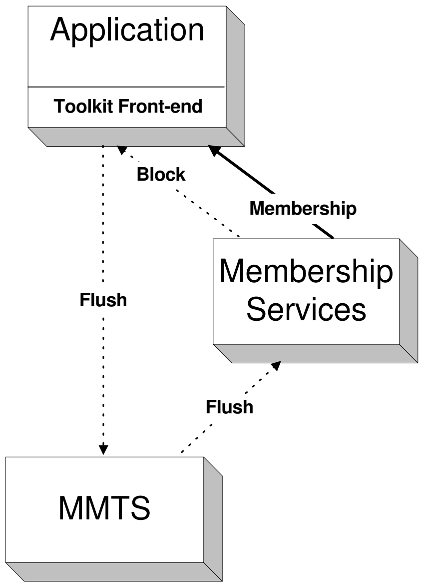

Figure 3: Supporting Virtual Synchrony

Note that after the FLUSH message, the application only stops sending messages for which virtually synchronous semantics must be enforced. The "bunches" concept of the MMTS [12], allows the application to continue transmission of messages for which virtual synchrony is not required.

An important innovation of Caelum is the support for hierarchical directory services. Caelum maintains a hierarchy of groups: a group may be a *sub-group* of a parent group. A parent group may contain a number of sub-groups. The listing of sub-groups and their memberships are available only to the members of the parent group. This concept is useful for applications containing a number of logically related groups, e.g., a conferencing applications with several discussion groups.

Caelum's membership service may maintain two membership sets for each process group: *active members* who may provide input in the group, and *passive members* who receive messages sent to the group but cannot send messages to the group.

4.2 Group Policies

Caelum allows users to determine *policies* regarding the membership and nature of communication in a group. The policies are declared when the group is created. If no policy is declared, then the policies are inherited from the parent group. There are two basic types of policies: *membership policies* and *run-time policies*.

Membership policies restrict the ability of processes to become members of the group. Restrictions may be imposed on the number of members in a group, and also on the properties of the members. For example, a conference over the Internet may allow only two members from each country to participate in the discussion. If due to a membership policy, a user's *join* request may currently not be fulfilled, Caelum allows the user to *block* until the *join* will become possible. Membership policies are enforced by the membership service of Caelum.

Run-time policies are specified at group creation time, and are enforced at run-time. Run time policies may define, for example, the number of users that may concurrently provide input in a group, and who is responsible for dispensing the right of speech. The floor control mechanism of Caelum (described in Section 5.1) enforces such policies.

Run-time policies are used in conjunction with membership policies. An example application that exploits both types of policies is an interactive chess server. The chess server allows two players to play in each game (actively join the "players" group), and allows other users to watch the game and exchange comments (passively join the "players" group, and actively join the "voyeur" group). Permission to play a game is granted according to the player's rank. Each of the two players may make a move only when it is his turn.

5 The Caelum's Session Services

Along with the membership and the QoS multicast, Caelum provides a wide range of services geared towards the needs of typical classes of distributed applications. Among them are tools for coordination and floor control in conferencing systems, consistent object replication, atomic commitment [34, 22], lock management, security *etc*. Let us consider some of these services in more detail.

5.1 Coordination and Floor Control

Different groups may impose different *run-time policies* on the eligibility of members to provide input of various types (e.g., video, audio, text) in a group. The policy is defined when the group is created. The floor control mechanism enforces this policy. An example policy may allow all the participants to type text concurrently in a text chat, and yet allow only one member to update a shared file at a given time. Another possible policy may designate a group of parties as the conference managers which are responsible for dispensing the right of speech. Ordinary members are allowed to speak only when they obtain permission to speak.

The floor control mechanism manages the dynamic switching of the right to produce input among multiple conference parties. This service is particularly useful in distance learning applications, in which students are typically not allowed to intervene when the teaching is in progress. Nevertheless, the teacher may grant students permission to ask questions at the end of a topic presentation.

The floor control of Caelum supports the *token* abstraction to designate a party (or group of parties) that are currently allowed to produce the input. The interface also allows parties to indicate their wish to obtain the token. If some participants were defined as conference managers they can pass the token among the ordinary parties at any given time.

If all the group members are equal in rights, Caelum allows them to freely compete for the token. The reliable totally ordered multicast service helps guarantee the uniqueness of the token holder. When the current token holder finishes its "monologue", he can explicitly pass the token to another party or return it to the system so that other parties can compete for it.

5.2 Support for Replication

Numerous distributed applications use replication in order to increase their availability and reliability. This arouses the need for a service that would preserve replicas in a consistent state despite network and machine failures. When the network partitions into several disjoint components, the states of disconnected replicas may diverge. When processes reconnect, all the processes should be brought to a common state.

Caelum provides two levels of consistency services: *short-term* and *long-term*. The *short-term consistency* service guarantees to preserve consistency within a group of connected processes. When a partition is mended, the states of previously disconnected replicas are unified using a state transfer protocol. The protocol in [4] exploits group communication for efficient implementation of state transfer.

The *long-term consistency* service guarantees a fully serializable history of object updates. This is achieved by prohibiting arbitrary updates of the object in disjoint network components, often, only the members of a *primary component* may update the object. The long-term consistency service can be implemented using a variety of algorithms with different availability guarantees. Group communication is a powerful building block for such algorithms, e.g., [23, 2, 3] all provide long-term consistency services using group communication as a building block. Caelum provides several types of *primary component services* that notify processes when they are members of the current primary component, e.g., a service based on dynamic voting [15].

5.3 Secure Group Communication

Caelum supports the notion of a *secure multicast group* [19, 32]. The communication in a secure group is encrypted. Each member multicasts messages that are encrypted using a secret key known to all of the group members. The access to a secure group is regulated by an authentication server. An authorized member is provided with a secret key for communication with other group members. Unauthorized members are prevented from multicasting messages to a secure group.

Hierarchical directory services allows secure groups to be hidden from unauthorized parties, by coupling them as sub-groups of the same secure parent group. Only members that are authorized to join the parent group may learn of the existence and membership of its sub-groups.

6 Applications

We presented the Caelum toolkit, which provides efficient solutions for all the communication requirements of CSCW and groupware applications. Furthermore, Caelum addresses all the aspects of sharing and replicating existing applications. Below we describe a few CSCW applications that we are currently developing using the concepts of Caelum. Similarly, Caelum may facilitate the development of distance learning applications, multi-party interactive games over the network, distributed simulations, and other collaborative computing applications.

6.1 On-line Conferencing

Multi-media and desktop conferencing systems are described in the survey of CSCW systems [30]. Such a system integrates several conferees (users), that cooperatively use a variety of applications in a shared work space. The shared applications have different QoS requirements:

Multi-Party Text Chat is simple to implement using group communication systems. We have constructed text chat applications both in C and in Java, using the Transis group communication system. These applications exploit the *causal* message service to guarantee that users see the answer to a message after seeing the message.

Video Conferencing requires high bandwidth low latency real-time message delivery, but also needs to exchange messages for connection establishment, flow control and *negotiation and renegotiation* of QoS (agreement on QoS parameters among the communicating parties) [28]. We constructed a video on demand service, [7], that exploits Transis to perform these tasks simply and efficiently.

Shared Drawing on a White Board requires low bandwidth *totally ordered* multicast, and *short term* consistency. A shared white board [33] was implemented in Java, using Transis.

Editing Shared Files also requires low bandwidth totally ordered multicast within a partition. However, in order to keep the replicated (or shared) files consistent, a *long term* consistency service is required.

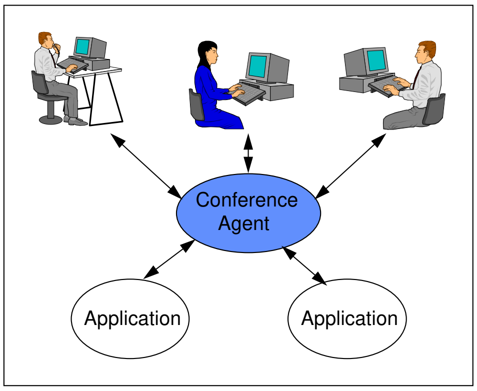

Figure 4: Desktop Conferencing Architecture

The *conference agent* controls the communication among the conferees and the applications. The architecture of a multi-media and desktop conferencing system is depicted in Figure 4. The distributed agent can exploit Caelum to provide the services listed in [30], e.g., floor control (i.e. input filtering), dynamic reconfiguration, consistent workspace replication and management, and logging the session. The hierarchical groups of Caelum support concurrent sessions and allow each user to choose which session group to join (provided that the group policy permits it).

6.2 A Jam Session Over the Network

In [18], Transis is exploited to conduct distributed jam sessions over a network. The musical notes are encoded in the MIDI [26] format, and are reliably multicast using Transis. Musicians that want to play together are organized in a Transis multicast group, and others may join as listeners. We intend to use Caelum hierarchical groups in order to conduct multiple sessions concurrently, and to allow users to choose from a variety of concurrent sessions. Caelum input filtering policies will be exploited to bound the number of musicians actively performing in each group.

References

[1] ACM. *Communications of the ACM 39(4), special issue on Group Communications Systems*, April 1996.

[2] Y. Amir. *Replication Using Group Communication Over a Partitioned Network*. PhD thesis, Institute of Computer Science, The Hebrew University of Jerusalem, Israel, 1995.

[3] Y. Amir, D. Breitgand, G. Chockler, and D. Dolev. Group Communication as an Infrastructure for Distributed System Management. In *International Workshop on Services in Distributed and Networked Environment*, number 3rd, pages 84–91, June 1996.

[4] Y. Amir, G. V. Chockler, D. Dolev, and R. Vitenberg. Efficient State Transfer in Partionable Environments. In *Proceedings of European Research Seminar in Advanced Distributed Systems (ERSADS'97)*, pages 183–191. Laboratoire de Systemes d'Exploitation Ecole Polytechnique Federale de Lausanne, March 1997.

[5] Y. Amir, D. Dolev, P. M. Melliar-Smith, and L. E. Moser. Robust and Efficient Replication using Group Communication. Technical Report CS94-20, Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, 1994.

[6] T. Anker, D. Breitgand, D. Dolev, and Z. Levy. Congress: CONnection-oriented Group-address RESolution Service. TR 96-23, Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, December 1996. Available from: `http://www.cs.huji.ac.il/labs/transis/`.

[7] T. Anker, G. Chockler, I. Keidar, M. Rozman, and J. Wexler. Exploiting Group Communication for Highly Available Video-On-Demand Services. In *Proceedings of the IEEE 13th International Conference on Advanced Science and Technology (ICAST 97) and the 2nd International Conference on Multimedia Information Systems (ICMIS 97)*, pages 265–270, April 1997.

[8] R. Baldoni, A. Mostefaoui, and M. Raynal. Causal Delivery of Messages with Real-time Data in Unreliable Networks. *Real-Time Systems*, 10, 1996.

[9] K. Birman, R. Friedman, and M. Hayden. The Maestro Group Manager: A Structuring Tool For Applications With Multiple Quality of Service Requirements. Technical report, Dept. of Computer Science, Cornell University, Ithaca, NY 14850, USA, February 1997.

[10] K. Birman and T. Joseph. Exploiting Virtual Synchrony in Distributed Systems. In *Symp. Operating Systems Principles*, number 11, pages 123–138. ACM, Nov 87.

[11] K. Birman, A. Schiper, and P. Stephenson. Lightweight Causal and Atomic Group Multicast. *ACM Trans. Comp. Syst.*, 9(3):272–314, 1991.

[12] G. Chockler, N. Huleihel, I. Keidar, and D. Dolev. Multimedia Multicast Transport Service for Groupware. In *TINA Conference on the Convergence of Telecommunications and Distributed Computing Technologies*, September 1996. Full version available as Technical Report CS96-3, The Hebrew University, Jerusalem, Israel.

[13] F. Cristian and F. Schmuck. Agreeing on Process Group Membership in Asynchronous Distributed Systems. Technical Report CSE95-428, Department of Conmputer Science and Engineering, University of California, San Diego, 1995.

[14] D. Dolev and D. Malki. The Transis Approach to High Availability Cluster Communication. *Communications of the ACM*, 39(4), April 1996.

[15] D. Dolev, I. Keidar, and E. Yeger Lotem. Dynamic Voting for Consistent Primary Components. TR 96-7, Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, June 1996.

[16] P. D. Ezhilchelvan, A. Macedo, and S. K. Shrivastava. Newtop: a Fault Tolerant Group Communication Protocol. In *The 15th International Conference on Distributed Computing Systems*, June 1995.

[17] R. Friedman and R. V. Renesse. Strong and Weak Virtual Synchrony in Horus. TR 95-1537, dept. of Computer Science, Cornell University, August 1995.

[18] D. Gang, G. Chockler, T. Anker, A. Kremer, and T. Winkler. Conducting Midi Sessions Over the Network Using the Transis Group Communication System. In *International Computer Music Conference (ICMC 97)*, September 1997. To appear.

[19] M. Hayden and O. Rodeh. Security in TransisE. Techical Report, The Hebrew University of Jerusalem. In preparation.

[20] M. Hayden and R. van Renesse. Optimizing Layered Communication Protocols. Technical Report TR96-1613, Dept. of Computer Science, Cornell University, Ithaca, NY 14850, USA, November 1996.

[21] ISO/IEC 13818 and ISO/IEC 11172. *The MPEG Specification.* `http://www.mpeg2.de/`.

[22] I. Keidar and D. Dolev. Increasing the Resilience of Distributed and Replicated Database Systems. *The Journal of Computer and System Sciences (JCSS) special issue on PODS 1995.* To Appear. Previous version in the 1995 ACM-SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), May 1995, pages 245-254.

[23] I. Keidar and D. Dolev. Efficient Message Ordering in Dynamic Networks. In *ACM Symp. on Prin. of Distributed Computing (PODC)*, number 15, pages 68–76, May 1996.

[24] C. Malloth and A. Schiper. View Synchronous Communication in large scale networks. In *Proceedings 2nd Open Workshop of the ESPRIT project BROADCAST (number 6360)*, July 1995 (also available as a Technical Report Nr. 94/84 at Ecole Polytechnique Fédérale de Lausanne (Switzerland), October 1994).

[25] S. McCanne and V. Jacobson. Vic: A Flexible Framework for Packet Video. In *Proceedings of ACM Multimedia*, pages 511–522, November 1995.

[26] B. Michael. *Music Through MIDI.* Redmond: Microsoft Press, 1987.

[27] L. E. Moser, Y. Amir, P. M. Melliar-Smith, and D. A. Agarwal. Extended Virtual Synchrony. In *International Conference on Distributed Computing Systems*, number 14th, June 1994.

[28] M. Rautenberg and H. Rzehak. A Control System for an Interactive Video on Demand Server Handling Variable Data Rates. In *Interactive Distributed Multimedia Systems and Services (IDMS)*, pages 265–276, March 1996. LNCS 1045.

[29] I. Rhee, S. Cheung, P. Hutto, and V. Sunderam. Group Communication Support for Distributed Multimedia and CSCW Systems. Technical report, Dept. of Mathematics Computer Science, Emory University, Atlanta, GA 30322, October 1996. To appear in ICDCS'97.

[30] T. Rodden. A survey of CSCW systems. *Interacting with Computers*, 3(3):319–353, 1991.

[31] L. A. Rowe, K. D. Patel, B. C. Smith, and K. Liu. MPEG Video in Software: Representation, Transmission, and Playback. In *High Speed Networking and Multimedia Computing, IS&T/SPIE Symp. on Elec. Imaging Sci. & Tech.*, February 1994.

[32] A. Rowley and J. Dollimore. Secure Group Communication for Groupware Applications. In *Proceedings of European Research Seminar in Advanced Distributed Systems (ERSADS'97)*, pages 222–227. Laboratoire de Systemes d'Exploitation Ecole Polytechnique Federale de Lausanne, March 1997.

[33] G. Shamir. Shared Whiteboard: A Java Application in the Transis Environment. Lab project, High Availability lab, The Hebrew University of Jerusalem, Jerusalem, Israel, October 1996. Available from: `http://www.cs.huji.ac.il/labs/transis/`.

[34] D. Skeen. A Quorum-Based Commit Protocol. In *Berkeley Workshop on Distributed Data Management and Computer Networks*, number 6, pages 69–80, Feb. 1982.

[35] B. C. Smith. *Implementation Techniques for Continous Media Systems and Applications.* PhD thesis, University of California at Berkeley, 1994.

[36] C. Topolcic. *Experimental Internet Stream Protocol: Version 2 (ST-II)*, October 1990. Internet RFC 1190.

[37] W. Vogels and R. van Renesse. *Support for Complex Multi-Media Applications using the Horus system.* Ithaca, NY 14850, USA, December 1994. On-line html document: `http://www.cs.cornell.edu/Info/People/rvr/papers/rt/novsdav.html`.

[38] B. Whetten, T. Montgomery, and S. Kaplan. A high perfomance totally ordered multicast protocol. In K. P. Birman, F. Mattern, and A. Schipper, editors, *Theory and Practice in Distributed Systems: International Workshop*, pages 33–57. Springer, 1995. Lecture Notes in Computer Science 938.

[39] R. Yavatkar. MCP: a Protocol for Coordination and Temporal Synchronization in Multimedia Collaborative Applications. In *Proceedings of the 12'th ICDCS*, pages 606–613, 1992. IEEE press.

[40] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A New Resource ReSerVation Protocol. In *IEEE Network*, September 1993. The RSVP Project home page: `http://www.isi.edu/div7/rsvp/rsvp.html`.