Net-based Learning and the mStar Environment

Kåre Synnes

Licentiate thesis
Institutionen för Systemteknik
Avdelningen för Programvaruteknik

1999:20 • ISSN: 1402-1757 • ISRN: LTU-LIC--99/20--SE
Net-based Learning and the mStar Environment

Kåre Synnes

Division of Software Engineering
Department of Computer Science and Electrical Engineering
Luleå University of Technology
S-971 87 Luleå, Sweden

May 1999

Supervisor
Ph.D. Dick Schefström, Luleå University of Technology.
Abstract

This licentiate thesis presents a Computer Supported Collaborative Work (CSCW) environment, mStar, from the perspective of net-based learning. The novel usage of IP-multicast based tools in teaching scenarios have been evolved to an interactive environment that supports time and location independence for students following distributed courses.

The mStar environment has been designed to be scalable through the use of IP multicast and a server-less design. Robustness is achieved by separating traffic by loss tolerance, where traffic that accepts no loss uses a reliable multicast protocol and traffic that can accept some loss may use repair techniques. To enhance robustness even further, network resource management is suggested (which is important within a corporate network). If the tools are to be used over a non-multicast enabled or low bandwidth network, then the traffic can be tunneled and even concentrated. Everything from small group meetings to large lectures is supported, which together with the possibility to use the tools asynchronously gives the flexibility needed. The tools in the environment are fully symmetric, which allows everyone equal access and thus supports full interactivity.

The mStar environment is well equipped to meet the requirements for net-based learning in the future. Students will have increased possibilities to take part of university level education. No longer are large geographical distances or time limitations ruling out where and when education can be offered. The student in year 2000 can be everything from a full-time student attending lectures physically at the university, to a part-time student following courses from his home at evenings and weekends. It might even be inexpensive, both from economical and time perspectives. Remotely attending students will also need to travel less, which further lowers the cost as well as the time spent. The student in the next millenium will certainly have demands on cost and time efficient education.

However, it is also important to offer a social context for a student. The creation of a virtual learning community is therefore suggested where students can meet and discuss subjects either by themselves or with a teacher involved. Video conferencing can create a virtual corridor, a virtual teachers' room, or a virtual group room for this end. The traditional lecture will have a less important role in education, but will still be the ‘pulse’ of a course. Instead more collaborative learning needs to be emphasized, such as group discussions, assignments and presentations. The traditional lecture will have to be enhanced with an electronic whiteboard, in order to make the lecture less static and more ‘alive’. A final belief is that remote students may become much more interactive than the physically attending students: they can interact using the whiteboard, chat or audio; and they can select to diverge into in-depth material and respond to issues discussed in the main session or side sessions like the chat. In some sense, it might even be better-than-being-there.
# Table of Contents

Publications .................................................................................................................. vii

Acknowledgements ................................................................................................. ix

Thesis Introduction ................................................................................................. 1

Paper A  Distributed Education  
using the mStar Environment ............................................................... 15

Paper B  Net-based Learning  
for the Next Millenium ................................................................. 33

Paper C  Robust Audio Transport  
using mAudio .................................................................................. 49

Paper D  The CDT mStar environment:  
Scalable Distributed Teamwork  
in action .................................................................................. 73

Paper E  Lightweight Application Level  
Multicast Tunneling using mTunnel .................................................. 93

Paper F  A Framework for Management and  
Control of Distributed Applications  
Using Agents and IP-multicast ................................................... 107
“The Best way to predict the future is to invent it.”

Alan Kay
Publications

This licentiate thesis consists of an introductory chapter plus six papers. The introductory chapter provides a coherent discussion of the issues in the six papers, and points out some future work in the area.


Acknowledgements

This licentiate thesis would not have been possible without the caring support by people around me. I cannot enough send thankful thoughts to my supervisor Dick Schefström for being a true visionary with the stamina to bring his ideas to reality, my resourceful colleague Peter Parnes for countless hours of help and directions, Johnny Widén for keeping his feet well stuck down in the soil as a centerpiece of stability, Ulrika Wiss for all the comments on my work using her keen eye for details, Fredrik Johansson and Håkan Lennestål for reading and commenting on my work, James Nord, Roland Parviainen, Per-Erik Björk and Jan-Åke Lehto for their support, and Svante Carlsson for believing in me.

I would also like to thank Ulf Bodin and Mikael Degermark for taking the time to comment on my work, Leif Kusoffsky for his vast efforts as a skilled teacher and colleague, Kristina Strandberg for all the caring support, Serge Lachapelle and my friends at Marratech for bringing our vision to commercial reality, and Mattias Mattsson for his sense of humor.

Special thanks to Mikael Börjeson for encouragement in times of distress. He mouthed “Men dom paket som kommer fram låter bra” (But the packets that arrives sounds good) in a comforting voice when the sound from my machine was utterly distorted by packet loss.

My work was sponsored by the Centre for Distance-spanning Technology, CDT, the European projects Mates (Esprit 20598), MediaSite (IA 1012 UR) and Roxy (Esprit 8410), and the project Education Direct, which is supported by the Swedish Foundation for Knowledge and Competence Development.

I would truly be a lesser man without my family and friends. Ruth, your loyal love and kind heart will always make my day brighter. Olav, your caring thoughtfulness and endless trust will always be my role model. You are the all perfect parents. Leiv, brother in blood and soul, with that special belief in me. I would be lost without you. My cousin Stefan, for being the third musketeer and a brother in heart. Karin, Per-Olof, Oskar, Elisabeth, Kristina, Ulrika, Alice, Kjell, Anna, Torbjörn, Susann, and all of you for your loving support. I would especially thank Mattias, a true brother in arms, for always being there. I could neither forget my friends in blood and sweat from the former A-group of Sollefteå Allmänna Sämsällskap for that special kinship, nor all of you that I have not mentioned.

Finally, I would like to thank the light in my life, my fiancée Magdalena, who holds my heart. Truthfulness, caring, love, energy, laughter. You make me complete.

Luleå, May 1999
Kåre Synnes
To Maggie
Thesis Introduction

1. Introduction
This licentiate thesis\(^1\) presents a Computer Supported Collaborative Work (CSCW) environment, *mStar*, from the perspective of net-based learning. The novel usage of IP-multicast based tools in teaching scenarios have evolved into an interactive environment that supports time and location independence for students following distributed courses.

The design goals have been to create an environment for net-based learning that is scalable for large groups, robust over lossy networks, flexible to meet different learning scenarios, and support interactivity.

This introduction will describe how the mStar environment is used for net-based learning at Luleå University of Technology, while describing the technology used for achieving the design goals of the environment.

1.1 Background
The Centre for Distance-spanning Technology (CDT) was founded in 1995, as a joint venture between Luleå University of Technology (LTU), Ericsson Erisoft, Telia Research and Frontec, with funding from the local municipality. CDT has since its foundation conducted research on net-based learning and collaborative teamwork environments. CDT is also conducting research in the fields of network and radio access technology. One of the results is the mStar environment [1,2], which includes a platform for implementing distributed applications based on IP-multicast [3].

The original vision was of a fully distributed environment for net-based learning and collaborative teamwork that offered a *better-than-being-there* experience for people not able to attend lectures, seminars or meetings physically. The mStar environment is today close to fulfilling the ideals of the ambitious slogan, where most of the tools and procedures required are developed and in use. One issue left to resolve is reducing the complexity of the environment in order to reach a broader mass with net-based courses using this technology.

\(^1\) *Teknologie Licentiate* is a Swedish degree between a *Master of Science* and *Doctor of Philosophy*.
Numerous courses have been given using the mStar environment, spanning from small informal graduate courses to full-fledged under-graduate courses with hundreds of participants [4,5,6]. The environment has also been in extensive use within most of the projects conducted at CDT for internal meetings and presentations. One factor of success has been that we live as we preach. The mStar environment has become an integral part of our daily work at CDT, both for education and for distributed work (from our homes and offices, which are geographically dislocated).

The deployment of the mStar environment has been made possible thanks to several interconnected IP-multicast enabled networks within the county of Norrbotten. Locally at the university, a campus network with 2000 students connected, has enabled many students to follow the courses from a distance. The county network has enabled local high schools to also participate in the courses given at the university, as well as giving own courses. A gigabit city network is under construction in Luleå and there are several initiatives for establishing high bandwidth connection to homes. This will further increase the number of people able to utilize net-based learning from their homes or companies.

It is therefore logical to foresee that a majority of the people in Norrbotten will have a high bandwidth connection to their home or work office within only a few years. This will impact most aspects of the society, among them how people will study. Assume that you can have learning content at your fingertips, and the choice to take part of it when it suites you. Would that not affect your *life long learning* ambitions? We believe this is the biggest incentive to conduct research on net-based learning.

2. Design Criteria

In a traditional radio or television setting for distributed learning, there is little or no support for *interactivity*, as these sessions are mainly focused on distribution. The student is therefore very limited to ask questions or take an active part in a learning activity. Later systems use video conferencing equipment based on ISDN H.323 services, but while they offer some interactivity they are based on expensive shared equipment and specially designed studios. For this reason, such studios are not commonly used for distributed learning. The last group of educational environments available today, is based on Internet protocols. Products such as RealNetworks RealPlayer [7] or Microsoft NetMeeting [8] are often used to convey content to students over the Internet, but these only give limited support for interactivity. The RealNetworks solutions are only focused on distribution, while NetMeeting have been designed for small groups. CU-SeeMe from White Pine [9] is also an alternative, but it is limited to audio, video and a shared whiteboard.

Thus, one major design criteria has been that all clients should be *symmetric* to allow all users of the environment, teachers as well as students, full interactivity. The clients have therefore been designed with IP-multicast as one major building block, which allow clients to effectively distribute any data to all other clients. By using a symmetric design based on IP-multicast instead of replication servers as in ISDN H.323 conferencing systems, many

---

2 One of these initiatives is the Internet Live project at CDT.
bottlenecks in scalability are also avoided. IP-multicast communication is based on a group address, where traffic is not directed to one receiver in particular. Instead, the network will forward the traffic to each group member with a minimum of data duplication. Each network segment will optimally only carry the traffic once and not, as with systems based on IP unicast or H.323, once per receiver on that network. This, together with the server-less design, makes the environment scalable for at least moderately sized groups.

Furthermore, the environment had to be robust. This is greatly alleviated by a server-less design, where there is no central shared server in the network for the clients to rely on. Also, the environment had to be robust enough to cope with single clients leaving the session without warning (due to network problems or even crashes). The environment had therefore to be designed as a loosely coupled service, meaning that all clients should be equal and fully distributed. No common state was to be kept in a shared location in the system.

As the Internet is a heterogeneous collection of networks, a certain degree of packet loss is always present. The environment should therefore be designed with packet-loss in mind, to make the clients more robust. This was achieved by dividing the traffic into two groups, one considered reliable and the other unreliable. Some clients, such as a chat client would use the reliable service, while other clients (such as the clients for audio and video) would use the unreliable service. As a special case, a whiteboard could be designed using both services, where intermediate changes uses the unreliable service and final changes uses the reliable service. The audio client also had to be able to repair packet loss in order to increase robustness.

The environment also had to be flexible enough to be able to adjust to varying usage patterns. A requirement was that the same set of components should be used for many different scenarios. This would make the environment easier and more consistent to use. New components could also quickly be designed when needed.

Another major design criteria was that a student at any time should be able to interact with fellow students or teachers. The environment should be accessible at any time from just about anywhere. The clients should therefore be designed to use inexpensive equipment on workstations or personal computers, and not rely on time-shared studios. It also meant that the environment had to be designed with asynchronous usage in mind, to allow students unable to take part of a live event to be able to view a recording instead. Furthermore, the environment had to have support for users located on non-multicast enabled networks.

The environment should also be designed with portability in mind, so that at least the most common platforms could be supported. Java in combination with C should therefore be used for implementation.

Finally, one downside of IP-multicast is that it lacks the ability to back off when congestion occurs in the network. Clients using much bandwidth therefore have to be able to adapt to current network conditions. They should also have support for external management of the available network resources.
3. Net-based Learning using the mStar Environment

The design and implementation of what would come to be the mStar³ environment started in 1995 at CDT. It was first designed around the usage of already available IP-multicast based conferencing tools like VIC [10] and VAT [11]. By 1997 a near complete set of tools for net-based learning and collaborative teamwork was designed and developed around these tools and work was ongoing to replace VIC and VAT with our own solutions. In 1998 a spin-off company, Marratech [12], was founded to commercialize the system. Note that most applications from Marratech have a history as advanced research prototypes at CDT.

This chapter describes the usage of the mStar environment at CDT for distributing course and seminar content, as well as introduces the various tools used. Note that the tools are not limited to net-based learning scenarios, as they are flexible enough to be equally usable for distributed meetings and other types of collaborative teamwork.

3.1 Audio and Video Conferencing

The need for real-time audio and video conferencing is inherent to most educational scenarios. There are simply few more efficient ways to communicate over a distance. It was natural that the before mentioned tools, VIC and VAT, was central to the development of an environment to support net-based learning and collaborative teamwork. However, the need for an own base of source code became evident, as tighter integration was needed in order to increase usability and to give room for experiments not otherwise possible. Both audio and video components have therefore been designed and implemented.

The extensive use of the mStar environment has shown that the most important of the different real-time media involved is audio, due to the fact that small disturbances easily can render the audio stream unintelligible. The video has mostly been used for achieving a sense of presence, and the other media are more or less non real-time (chat, whiteboard) since they use a reliable protocol for transport. Efforts have therefore been spent to study how to achieve the best audio quality during different network conditions.

An audio tool, mAudio, has therefore been designed together with a tool for subjectively measuring perceived audio quality [13]. The results from the initial evaluations show that up to 20% packet loss can be tolerated when simple recovery techniques are used. The subjective tests of audio quality was conducted using 2, 5, 10, 15, 20, 30 and 40 % loss rates with silence substitution, noise substitution, single redundancy, single repetition and double repetition as recovery methods. This pilot evaluation confirmed the results from other investigations, and is a good base for future evaluations on more advanced recovery techniques.

³ The name mStar comes from the usage of a small m (for multicast) in all tool names. Using the wildcard, ‘*’, became a common way to rely to all of the tools, hence the mStar environment.
The subjective tests are based on a three-sample technique, where one sample has the full original quality, one is distorted with 40% loss and repaired with silence substitution, and the last one is the sample with random loss and recovery technique. The latter sample is then subjectively compared to the other two by the subjects. This technique avoid the pitfall of a rather uncertain lower limit of quality used in traditional tests, where only two samples are used and the lower limit is described as 'significantly distorted'.

One algorithm, used in the mAudio tool, is also presented that gives a reasonably good loss tolerance under network conditions with moderate packet loss (up to 20% loss). Creating IP-based groupware applications, such as net-based learning environments, that are resilient to loss, delay and delay variation is therefore possible.

The mAudio component is today integrated with video components in Marratech PRO mVideo. Figure 1 shows mVideo with the mAudio component integrated.

3.2 Distributed Lectures and Seminars

Presentation material is prepared in advance of a lecture by using SlideBurster [14], which is a tool for generating multiple slides from one HTML document. The slides are then distributed using mWeb [15], and then presented in a web browser. An integrated mWeb client and a web browser, Marratech PRO mViewer, are currently used for distribution of presentations at CDT. A whiteboard component, Marratech PRO mWhiteboard, is also available as a substitute for the traditional board. mVideo, mViewer and mWhiteboard are together with a touch-sensitive SmartBoard [16] and a projector, a very effective environment for distributed presentations.

Optionally a textual chat tool (mChat), a voting tool (mVote) and a floor control tool (mWave) can be used to enrich a session. A tool for remotely controlling cameras, Director, is also available. These tools are further described in [14].

3.3 Recording, Editing and Playback of Media

The requirement for asynchronous usage of the mStar environment is met by the mMOD system [14], which can record and replay any session. mMOD is based on a simple web interface with a Java applet for controlling the playback of a session. A prototype for editing of recorded sessions, mEdit [17], is also available. Noteworthy is that each slide
sent using mWeb could be used to generate an index in the recorded session, so that a specific point can be found easily when played back. It is also possible to add indexes manually.

3.4 The Virtual Learning Community

The traditional lecture mainly offers a one-to-many channel, since students often are too shy to ask questions – especially if the session is large. We have found that something complementary is needed, where students can cooperate and interact with each other. It is naturally very important, perhaps even vital, for remote students to have a continuous contact with the teachers and fellow students.

Today students themselves are sources of information, the teachers are no more the sole information source, which means that collaboration and sharing of information is vital to make large distributed courses a success. Creating a learning community can also lessen the workload for both students and teachers.

However, most of the courses currently given using the mStar environment at LTU are ‘normal’ university courses. This means that the incentive to establish a virtual student community is weak, since there is a physical community to fall back on. It is always possible to knock on the teacher’s door. The result is also that the current virtual student communities are sparsely populated. This would change if most students were ’real’ remote students, and the physical community was not available.

It is possible to create virtual teachers’ rooms by using a session where the teachers of a course are available to answer questions or chair discussions. This could also be a session where a group of teachers are available – thus not necessarily linked to a specific group, but as a general resource. The latter case might be important for the feeling of continuity for remote students, since they change courses every 10 weeks and need a fixed point during their studies.

A session can also be used by a group of students, forming a virtual group room, to discuss course related issues, with or without a teacher attending. The virtual group room can also be used for project status presentations, where a subgroup of a larger class is attending, and to discuss and view recordings. This enables interactivity even during playback of recordings, when the playback can be paused and discussed freely. Naturally a teacher can be invited to further enrich the discussion, perhaps while using additional tools.

This form of meetings may prove invaluable, as remote students need to overcome isolation by forming groups and socialize. It is therefore important that net-based courses emphasize collaborative over individual learning, in order to stimulate use of this media.

The value of asynchronous communication should not be underestimated, as many students choose to follow courses entirely or partly asynchronously. Systems that support asynchronous communication are numerous, ranging from simple mailing lists to advanced WWW-based board systems. A few divisions at LTU are using a system developed
internally, World Wide Web Communication System - W3Cs, which enables students and teachers to communicate via a web-based bulletin board\textsuperscript{4}. W3Cs also offer basic support for course management (administration of students), a task earlier handled by homemade scripts individual to each teacher and course, as well as support for document publication.

One observation made is that usually silent students also contribute to asynchronous communication, maybe because they have more time to formulate their meaning or since the interaction is more abstract. This is more noticeable if an electronic group is maintained over time, creating a virtual community of students where the students 'feel like home'. The virtual community must however have an active leadership (introducing new members and excluding misbehaving members) and clearly defined boundaries (limiting what should be discussed).

3.5 Lessons Learned
The statistics from our mMOD server logs show that many students prefer to watch lectures during evenings, or even late at night. This is confirmed in [18], which studies the logs from mMOD. That study shows that replay of educational content have peeks in the evenings (often in conjunction with homework assignments or exams) and that it is common that only partial replay is done, suggesting that the content is browsed for some particular information. The possibility to watch recordings is thus clearly useful for students having overloaded daytime schedules. Using the playback facilities offers another clear advantage: it enables students to take pauses, to either read additionally related information or to consult the course literature. Unfortunately, these students cannot be part of the spontaneous discussions during lectures. Having multiple participants active in the playback environment might remedy this to an extent.

We have noticed that other social protocols have been established when using the environment for presentations and education. Foremost are the sub-discussions that take place using the chat tool, where a set of the participants either discuss the presenters material or something completely uncorrelated. This kind of discussions and sharing of information enhances the learning experience, since attending a lecture physically normally disallows side conversation in the audience.

By encouraging the use of different means of communicating electronically, such as email or WWW-based discussion media, we have found that students tend to help each other. This form of social clustering is most interesting. It lessens the traditional burden of a teacher, where students with additional knowledge often share it with the rest of the class and the teacher. The fact that students are able to share this knowledge with the group is an enormous advantage to more traditional teaching, where students seem to rarely form groups with more than five members.

A downside is that lectures distributed with the mStar environment tend to become more static than classical (i.e. non-electronic) lectures. Experienced teachers are most often those

\footnote{The tool W3Cs have nothing to do with the World Wide Web Consortium, W3C.}
who can improvise and dynamically alter the course of a lecture. These teachers usually do not need to prepare presentation material, as their lectures often take the shape of a normal conversation. With mStar, teachers are easily 'caught' in the flow of their pre-prepared electronic material. It is therefore very important to still allow the teacher to improvise, perhaps by making use of an electronic whiteboard or a sketch board.

### 3.6 Lightweight Application Level Tunneling

Since many networks still do not support IP-multicast there is a need for an application that tunnels IP-multicast traffic. mTunnel [19], is a lightweight application level tunneling application where the end-user controls which MBone-sessions and which IP-multicast groups to tunnel through a Web-interface. To save bandwidth, tunneled streams can be transcoded on the data-level; and traffic sent through the tunnel can be compressed by grouping several packets together and using statistical compression. This may reduce the bandwidth sent over a tunnel with between 5 and 14%.

The mTunnel application can also take more intelligent measures in order to lower the consumed bandwidth over a low bandwidth link. For audio it can mix all sources into one and then recode it to a lower encoding, such as GSM. It can also use the information about whom that are sending audio and filter all video sources but the corresponding one, which can be rate manipulated as well to further lower the bandwidth consumption.

### 3.7 Management and Control of Distributed Applications

When a distributed desktop conferencing application, such as the mStar environment, is deployed in a large organization; a number of new management and control issues evolve since the IP-multicast traffic cannot be allowed to disturb other services in the network. The administrators of the network therefore need to get information about and control:

- which users are part of which conferencing sessions,
- which media in each session do they currently have active, and
- if the user is currently transmitting any data within a session and then with which settings.

If this information is available it will allow the administrator to control both session membership (e.g., kick out unauthorized members) and the total bandwidth used by this group of applications. This means that the network administrators can control the total amount of bandwidth used by each user and session explicitly. A framework and a prototype application, mManager, has therefore been designed and implemented [20]. The target of the framework is to allow for resource discovery of both controllable elements and available control points in these elements as well as real-time control.

### 4. A vision of the Future

Our belief is that students will not be much different than today, but that they will have increased possibilities to take part of university level education. No longer are large
geographical distances or time limitations ruling out where and when education can be offered. The student in year 2000 can be everything from a full-time student attending lectures physically at the university, to a part-time student following courses from his home in the evenings and weekends. It might even be inexpensive, both from economical and time perspectives. By giving courses for a larger number of students, they will become cheaper per student. Remotely attending students will also need to travel less, which further lowers the cost as well as the time spent. The student in the next millenium will certainly have demands on cost and time efficient education.

By combining large-scale distributed lectures with group projects and personal assignments, well-balanced distributed courses are attainable. The lectures should contain invited speakers from the industry or other research organizations, to complement the traditional academic content and to better prepare students for what is required after graduation. The group projects leads to an early training of social skills, while personal assignments remain important for grading reasons. However, when courses are attended by a large number of students, it will be increasingly important to establish personal tutors or advisors. Personal relations to teachers are important for the overall learning experience, students simply need someone to ask questions.

The future of net-based learning includes equal parts of freedom and responsibility. We can clearly see that the old way of spoon-feeding students with information is coming to an end, and a more collaborative shared learning experience is forth growing. This however needs students with high self-motivation, discipline and commitment. It also put higher demand on the courses, on content as well as pedagogy. The new net-based courses must apply specially developed content, following the old tracks simply do not work, and new ways of teaching is also required, which emphasize collaborative learning.

4.1 The Future Lecture Hall

The lecture hall will of course continue to play an important role for net-based learning, both for remotely and physically attending students. We believe that net-based lectures will be the 'pulse' in distributed courses, that controls the rate of which the major part of the students will follow the course.

There is however a need to allow more freedom for the lecturer to improvise and break free from the pre-prepared material, such as a slide show. The vision is to combine the already existing tools with an electronic whiteboard, onto which some of the tools are projected and manipulated with a pen by the lecturer.

Figure 2, The Lecture Hall
Another device is a tool for navigating the slide show remotely, thus unlocking the lecturer from the position behind the lecturer station and allowing him to move more freely about. This clearly makes the lecture more alive; a lecture where the speaker stays at one position and just turn slides is quite unattractive. Figure 2 depicts a lecture hall using: A – a shared whiteboard with pen support, B – a projected virtual corridor, C – projected web slides, and D – a lecturer station.

The whiteboard is shared so everything the lecturer draws on it will also be visible for the remote students, and what the remote students draw will be visible for the lecturer and the students in the lecture hall. The pen-based input mechanism is needed since drawing with a mouse has certain drawbacks: it is hard to draw precise shapes; the lecturer is 'locked' to the computer and is not allowed to move around and draw, which gives the lecture a more static impression; and using the pen is more similar to 'normal' lectures. Using a digital whiteboard, such as the SmartBoard, is therefore compelling. It offers direct manipulation of not only a whiteboard application, but also other applications.

A virtual corridor with the remotely attending students is projected on a screen for giving the physically attending students the feeling of presence. We have previously seen that students tend to be very shy, which might be due to that there are unknown participant and that they feel uncertain because of that reason. The visible presence might lessen this effect and yield increased interaction within the lecture hall. An alternative could be to place the virtual corridor along one of the sides, together with a speaker for sound from the network. This might further lessen the shyness as the remote students are ‘at the same level’ as the physically attending students. The differentiation of the sound sources would also help the people in the lecture hall to position the speaker (the lecturer’s voice would be coming from speakers in the front of the lecture hall).

Note that the remote students could be able to interact more than the physically attending students: they can interact using the whiteboard, chat or audio; and they can select to diverge into in-depth material and respond to issues discussed in the main session or side sessions like the chat.

The current use of the mStar environment yields fairly static lectures, which many students claim are dull and not inspiring. Using more than one media during a lecture remedies this to some extent. That is, using the browser for presentation of pre-prepared slides and the whiteboard for drawing interactively. This together with discussions and small exercises will make the lecture more ‘alive’. It can be as simple as just asking a question and give the students a few minutes to discuss it in pairs. The remotely attending students can discuss the same question using the chat tool. Then the question is discussed by all to reach a common conclusion.

Lastly, student seminars are good examples of when students are well activated and really have the chance to share knowledge. Note that CDT has a strong emphasis on technology and not on pedagogy. There therefore exist a need to focus on pedagogy in order to evolve the environment, in corporation with pedagogues.
4.2 The Digital Library
The library has throughout the centuries been the central institution at most universities, where students have roamed the vast content of recorded information. Today there is a need for a digital library to take the place as the central base for net-based studies.

Several efforts are ongoing for creation of such digital libraries, where not only written information (contained on physical paper or on digital media) is to be stored. The benefits of the digital libraries are obvious:

- Information can be retrieved independent of time and space.
- Everyone have equal access to information.
- Information can be shared or distributed over several digital libraries.
- Information agents can make it easier for the student to select information.

During four years a large number of lectures and seminars have been recorded and placed on a media server at CDT. This example, that is used on a daily basis by students, show that the extension of a digital library is both feasible and useable. We therefore foresee the development of digital libraries containing searchable multimedial content.

This will further release the students from the bond of time and space, enabling net-based and life-long learning not only for a few but also for the majority of students. The efforts to modularize and modernize net-based courses, while placing pedagogy in the first rank, make us believe that net-based learning has come to stay.

5. The Education Direct Project
The Education Direct project was initiated in 1996 as a major effort on exploiting new technology for net-based learning. The goals were to:

- Accomplish a significant increase in broad use and understanding of multimedia technologies on the Internet in the county of Norrbotten.
- Establish better direct contacts between CDT / Luleå University of Technology, and high schools and secondary schools, and small and medium enterprises with respect to this technology.
- Illustrate the use of the technology by distributing a course on this actual subject (Internet and Distributed Multimedia).
- Establish and test the infrastructure needed to accomplish this.

This as the first concrete step to implement the vision of the net-based presentation and learning environment, according to which most courses at the Luleå University of Technology and CDT will be possible to attend independently of physical location. This is a need that is especially urgent in the county of Norrbotten, with its sparse population and large geographic distances.

In the first phase of the project an undergraduate course in Internet and distributed multimedia was distributed. The participants came from all over the county, many of which
were secondary-school teachers, (as those will act as local technology transfer persons) and local under-graduate students. The local students had the choice of following the course by attending the lectures physically or virtually using the mStar environment.

6. Organization of the Thesis
This licentiate thesis consists of an introduction and six papers, that gives an overview of how net-based learning is conducted at CDT together with a description of some of the tools involved.

Paper A, “Distributed Education using the mStar Environment”, presents the initial results from using the mStar environment for net-based learning.

Paper B, “Net-based Learning for the Next Millenium”, build on paper A and presents a future vision of net-based learning together with conclusions drawn from extensive usage of the mStar environment for learning scenarios.

Paper C, “Robust Audio Transport using mAudio”, present the design and implementation of the mAudio application together with initial results from a subjective audio quality evaluation.

Paper D, “The CDT mStar environment: Scalable Distributed Teamwork in action”, presents the generic agent-based structure of the mStar environment together with many of the applications in the environment.

Paper E, “Lightweight Application Level Multicast Tunneling using mTunnel”, presents the design and usage of the mTunnel application.


7. Contributions
My work has been focused on the audio component and learning issues of the design, implementation and usage of the mStar environment. The novelty lies in the construction and deployment of this environment. I have also been involved in the design and implementation in general.

Papers A-C are mostly my work, while Peter Parnes has played a major role in papers D-F.

8. Summary and Conclusions
The mStar environment presented in this thesis is scalable through the use of IP-multicast and a server-less design. Robustness is achieved by separating traffic by loss tolerance, where traffic that accepts no loss uses a reliable multicast protocol and traffic that can accept some loss uses repair techniques. To enhance robustness even further, network
resource management is suggested. Everything from small group meetings to large lectures is supported, which together with the possibility to use the tools asynchronously gives the flexibility needed. The tools in the environment are fully symmetric, which allows everyone equal access and thus supports full interactivity.

Students using the mStar environment are no longer constrained by physical distance, they can easily take part of lectures, seminars and group discussions electronically. They can also be connected via non-multicast enabled low bandwidth networks, since they can tunnel concentrated traffic to their local hosts. The students are also less bound by time, as they can use a mMOD server to access recorded sessions or the WWW to access course information.

9. Future Work
In order of increasing usability of the mStar environment for learning scenarios, it would be good to have a specialized and simplified client. It also has to become simpler to use the other parts of the system, from generating presentation material to manage recordings.

A number of courses have been given using the mStar environment, but there has not been any real investigation done about the effects of using it. This is the logical next step, where two groups of students could be used. The first group would follow the course in the traditional way, while the second would have access to the mStar environment as well. This would yield a number of interesting questions to pursue:

- How many of the students actually use the tools?
- How is the environment actually used?
- How many thinks it is an added value?
- What is the study result between the two groups?
- How much time is spent to learn the content of the course?
- How could the environment be made better?
- How are the study pattern changed?

Another area of work is to integrate a course management tool into the environment, such as the Luvit system [21]. The Luvit system is more functionally complete and mature in comparison with the W3Cs developed at LTU.

References
Thesis Introduction


Paper A

Distributed Education using the mStar Environment
Distributed Education using the mStar Environment

Kåre Synnes, Serge Lachapelle, Peter Parnes, Dick Schefström
Luleå University of Technology / Centre for Distance-spanning Technology
Department of Computer Science
971 87 Luleå, Sweden.

ABSTRACT

The mStar environment for distributed education utilizes the WWW and IP-multicast to enable teacher-student collaboration over large geographic distances. Several educational projects, spanning from secondary school courses to company internal training, have deployed the mStar environment.

This paper reports on experiences gained over a year of practice at the Luleå University of Technology and the Centre for Distance-spanning Technology. The paper presents the methodology and technology used, while recognizing usage scenarios such as preparation of presentation material, distributed presentations, asynchronous playback of recorded and edited material, and virtual meetings for educational support.

Keywords: distributed education, mStar environment, real-time, MBone, WWW.

1. Introduction

The WWW community's strive for content quality has created a quiet revolution in education. In fact, much work in this field has been presented at past WWW conferences. The many efforts related to the educational uses of the WWW [1,2,3] and virtual classroom environments [4] have been a major influence for this revolution.

The availability of course related information such as lecture notes, extra course material, exercises, and course scheduling blended with the WWW's inherent qualities such as hyperlinks and accessibility have added much information to the classical structure of courses.

Although undeniably useful and valuable, education on the WWW has lacked a fundamental feature: the presence of quality video and audio for natural spontaneous interaction. WWW-based solutions such as 'HTML courses' for 'electronic-education' have somewhat restricted the exchange of information between students and their teachers. More recent technical solutions, such as the use of multimedia in WWW documents, are limited

1 This is an extended version of a paper presented at the WebNet '98 conference in Orlando, Florida. The paper received a "Top Full Paper Award".
to simple playback control, thus leaving no room for spontaneous interactivity. This deficiency has prevented broader use of distance education on the WWW, since university courses should offer the opportunity for discussions and debate.

This paper reports on more than a year of research and actual usage of the mStar environment [5,6,U1] in projects aiming to use and demonstrate the full potential of distributed multimedia education. It will first present a brief background, then put forward different usage scenarios and tools, and finally provide a detailed discussion about experience acquired from usage of this new education environment.

1.1 Background

Bringing quality distance education and collaboration to the Internet is one of the driving forces behind the Centre for Distance Spanning Technology [U2], CDT, at the Luleå University of Technology [U3]. The university is located in the county of Norrbotten (see figure 1), which consists of the northernmost fourth of Sweden and covers approximately 160 000 square kilometers (62 000 square miles). The population is sparse, with about 260 000 people.

This has meant that many high schools cannot gather the critical mass and competence to offer the courses and subjects that are possible in the more densely populated areas of Sweden. By giving WWW-based courses over the networks, a sufficient critical mass is generated, creating a countywide virtual university with breadth and quality that might otherwise not be possible. The effects on society and the region could be great, as primary and secondary schools in the county collaborate with the university using this new technology for distributed education.

Furthermore, the funds per student at the Luleå University of Technology are continuously decreasing. During the last three years, from 1995 to 1998, we have witnessed a decrease in funding of 15%. The resources left available will have to be used more efficiently. The normal way to compensate for funding cuts is to create larger student groups. An efficient solution to manage these bigger groups of students is having a more teacher-independent 'virtual student community', where students can collaborate in solving problems. This may reduce a teacher's increased workload due to bigger classes.

Giving WWW-based courses and creating a virtual student community is made possible thanks to a unique Internet engineering project, IT Norrbotten [U4], which has built a multicast enabled high-speed network infrastructure between communities and companies in the county. Together with the university campus network (connecting about 2000 student
apartments), this has created an excellent communication framework for distributed education.

The Luleå University of Technology has given a number of courses using the mStar environment, ranging from graduate-courses to full fledged under-graduate courses. The first course using the technology was about the technology itself, Distributed Networked Multimedia [U5]. About 110 under-graduate students followed the course together with an additional 30 students from the county area. Other under-graduate courses have been given using the same methods, such as a course in Object-Oriented Programming [U6] with more than 120 students. All of the graduate courses at CDT [U7] have been conducted using the mStar environment as well. Therefore, the University has achieved a significant deployment and usage of distributed education over the Internet.

Today many large companies, such as Telia [U8] and Ericsson [U9], are showing a growing interest in the technology as well. Several courses for the companies have been given using the technology, and the Ericsson deployment is progressing rapidly. Giving joint courses might help bridge the gap between local industry and the university. At Ericsson-Erisoft [U10] (which has 560 employees in Norrbotten), many workstations are capable of running the mStar environment. mStar is used for courses and presentations as well as traditional meetings, thus reducing the need for travelling between the three offices.

This paper therefore presents the concrete results of a wide deployment effort of the mStar environment for distributed education where secondary schools, the university, local companies and communities are all active participants. By now a large amount of persons have tried the mStar tool suite for education, with varying degree of satisfaction. We are now only starting to see the first social and cultural changes within the schools and companies involved.

2. mStar Distributed Education Scenarios

The mStar environment is used in a number of education related scenarios, which today is used to give real-time interactive courses throughout the county of Norrbotten. Presenting these scenarios offers a perfect opportunity to put the reader in the context of distributed education and to introduce the mStar environment.

2.1 Preparation of Presentation Material

The first scenario is one of preparation. It mainly revolves around the preparation of a lecture's content. This step involves the preparation of traditional presentation material using HTML (see figure 2).
The benefits of HTML for an overhead medium are numerous:

- Traditional WWW hyperlinks that point to more information can be inserted in the slides.
- Users viewing these slides on their desktop computer can control the document's window size, font sizes and colors through the browser's preference settings. This can greatly help people with viewing disabilities.
- HTML is a very portable format that is widely supported across numerous platforms for both viewing and printing.
- Sending HTML slides using multicast uses very little network bandwidth in comparison with filming the slides.

With the help of SlideBurster [U11], the teacher can divide a single HTML document into a number of different slides. The tool automatically creates links to each of the slides and creates an outline for the lecture. In addition to creating separate slides, properties such as colors, logos and author information can easily be formatted. Once the slides are ready, the teacher can publish the slides on the course's WWW pages before each lecture. Overall, this step helps the students to prepare for lectures as well as enhances the quality of the class material thanks to the many hyperlinks and pictures of related material.

2.2 Distributed Presentations

Once the course material prepared, we can now proceed to a scenario involving the actual lecture. For this to be possible, the teacher or a class technician must go through a certain number of steps.
1. For students to 'tune-in' to the lecture, the MBone [U12] session must first be created and announced on the WWW. This is done via the WWW-based session directory mSD (multicast Session Directory, see figure 3) [5:p.4], and mAnnouncer (multicast Announcer) [5:p.4].

2. Once the different media sources are being transmitted, a tool called mVCR (multicast VCR) is used to start recording on the mMOD (multicast Multimedia on Demand) server [5:p.7].

3. During the lecture, the technician can remotely control positions, zoom and focusing of the two cameras inside the lecture hall with the help of mDirector (multicast Director) [5:p.9]. The cameras are used together with video grabbers to digitally capture the audience and the teacher. The audio and video streams are sent throughout the network using IP-multicast [7].

The students can 'tune-in' to the appropriate lecture by pointing their browsers to mSD's WWW page [U13]. The main purpose of mSD is to present an interface to all available sessions. From mSD students can launch all the proper tools, such as VIC (Video Conferencing Tool, see figure 4) [U14] for video, mAudio (multicast Audio, see figure 5) [5:p.5] for audio as well as the other mStar tools.
This simple step is critical since only limited technical knowledge should be required to fully take part in a session. Hence, a lecture is never more than “a few clicks away”.

The participant is then 'submersed' in an environment that takes distance education a step further from traditional HTML-based courses. The student is no longer a passive receiver as he can interact in real-time. Students participating physically in the lecture hall can hear questions asked by online participants through the audio system and see the online participants through a projection on a wide screen. Naturally, all other on-line participants also hear them. This creates a very symmetric environment for two reasons:

1. Every participant, including the teacher, has access to the same facilities. Everyone can participate equally in the discussion. We feel this is a very important feature for promoting student participation and debates between class members.
2. The delivery of all the multimedia content is achieved through IP-multicast, which substitutes the traditional client-server structure for a symmetric method of delivering multimedia content.

As the lecture progresses, mWeb [5:p.39,8] is used to synchronize the teachers' WWW browser with all the participant's browser windows, thus working as a distributor of presentation material. This greatly improves the overall ease of use as well as the lecture's natural flow for the on-line participants. The mWeb is an important part of the environment; therefore it is extensively described in section 3.

Meanwhile, a participant can interact with the teacher and the other participants by raising his hand using mWave (multicast Wave, see figure 6) [5:p.17 (was mW2T)], thus imitating the social protocols of a normal classroom. Participants can also use mChat (multicast Chat) [5:p.6] and mWhiteBoard (multicast WhiteBoard) [5:p.6 (was mWB)] to discuss issues with other on-line students without interrupting the lecture or to part in lecture exercises. Interaction can also take the form of voting on different issues by using mVote (multicast Vote) [5:p.6]. This gives on-line students possibilities that do not exist in a classic classroom environment.

![Figure 6, mWave, multicast Wave](image-url)
Furthermore, the teacher can include a playback of a recorded session into the live lecture, which enable reviewing and debating of related recorded material.

2.3 Asynchronous playback
The lectures are recorded using the mVCR application and then edited using mEdit [U15]. Indexes, i.e. named temporal points in the lecture, can be added by the technician while the lecture is taking place or by the teacher afterwards. Adding indexes involves using mIndex [U16] and mEdit. A teacher can also add comments, modify the flow of events, remove sequences such as a long pauses and insert previously recorded multimedia content. Adding slides, a famous speech by a Nobel prize winner or a clip from a previous lecture can easily add a lot of value to a lecture’s content.

The WWW interface to the mMOD server allows reviewing recorded lectures by starting playback sessions. Participants can join in on playbacks currently being played by others or start their own playback (see figure 7). Interaction between the participant and the mMOD server is done via a mVCR control-applet started from the mMOD WWW page (see figure 8) [U17]. mVCR provides basic VCR-like functions and access to the indexes of the lecture. It enables the student to quickly jump to the desired part of the lecture without having to fast-forward through the lecture. During playback, the participants can view all multimedia sources and events that occurred in the original lecture. The flow of the slides, mChat, mWhiteBoard and mVote events are all preserved and played back.
2.4 Virtual meetings

Aside from lectures, using this environment in combination with newsgroups and traditional mailing list can create a 'virtual student community' in which students can help each other for labs and participate in course related discussions. Students are able to cooperate and interact with each other using the previously mentioned suite of tools. Helping other students with labs, course questions or simply sharing experiences add a collaboration dimension to distance based courses. Creating such a community, as described in [4], can be very useful for both students and a teacher's workload.

It is also possible to have a 'virtual teachers room' session using audio and video tools. This works like a virtual corridor, where the students enter and ask questions or discuss course-related issues. For distant students it is naturally very important to have a continuous contact with the teachers.

By combining the possibilities offered by available networks, the collection of portable tools written in Java, the accessibility and ease of use of the WWW and the benefits of IP-multicast, we have been able to make these scenarios part of our everyday, real-life teaching experiences. We would like to stress that this is a working system in real use.

3. The mWeb Application

As the distribution of the WWW based presentation material is very central in the mStar environment for distributed education the following section is devoted to further explain about the mWeb application.

The mWeb application is a tool for real-time distributed presentations with HTML as its presentation medium. The application includes functionality for distribution of HTML-pages, including in-line data and embedded objects, pre-caching of files to be used within a session, on-demand fetching of files, synchronization between browsers, and interfacing different WWW browsers. mWeb uses the mDesk framework for distribution and control [5:p.23,9].
The problem of adding real-time distribution of HTML to the WWW can be divided into two parts, synchronization and distribution. This section discusses the architecture of the mWeb application and how these problems have been solved in mWeb.

3.1 The Architecture

The mWeb application acts as a gateway between a WWW browser and the MBone (see figure 9), mediating distribution of HTML-pages (see section 3.2) and 'display-messages' (see section 3.3). The application can also run in a so-called lightweight mode, where only the URLs to be displayed are multicasted. This is useful in smaller groups as the delay becomes shorter and the network usage does not significantly change.

![Figure 9: The mWeb Architecture](image)

HTML-pages to be displayed during a session can be collected in three ways:

1. URLs to be displayed, including URLs to any inline data, are specified manually in a file by the presenter. This file is then used by mWeb for the distribution of the data to be presented.

2. URLs are collected dynamically during a presentation using a browser that supports the Common Client Interface (CCI, currently only supported by the XMosaic browser) [U18]. This means that whenever the presenter selects a link or changes HTML-page (for instance using the history in the browser), information is sent from the browser to the mWeb application.

3. URLs are collected dynamically during a presentation using the special mWeb WWW-proxy that sends information about the requested pages to the mWeb application. This is achieved by directing the browser to request all pages through the proxy, instead of fetching them directly. Unfortunately, this create problems when using HTML frames as mWeb interpret this as several quick requests (an HTML frame-page may consist of...
several HTML files). To solve this, mWeb tries to guess if it is a frames page based on the URLs requested and the time between the requests.

Another way of solving this would be to let mWeb parse each requested HTML-file and take proper actions when a frame-page is encountered. However, the overall advantage does not justify the overhead of introducing an HTML-parser into the application.

The last method is the one most commonly used as it is the method (out of the three presented) that puts the least burden on the presenter before and during a presentation. It allows a presenter to distribute the presentation material to the listening group members without them even being aware of what is actually being done.

During the presentation a window containing a list of displayed pages is shown (see figure 10). At the presenters side the list will contain all pages, but on the listeners side only pages that have already been displayed are listed.

**Figure 10: mWeb, multicast Web**

### 3.2 Distribution of Presentation Material

The first problem related to distribution of the WWW based presentation material is how to distribute WWW pages efficiently to a large group of listeners. The simple solution would be to let each receiver fetch the page to be displayed directly from the WWW server. This would unfortunately create a large burden for the server if the group were large as all listeners would request the same page at nearly the same moment.

Instead, the presenters mWeb instance fetches the page content to be presented from the server and then distributes it to the listeners. The distribution is done using the /TMP (Tunable Multicast Platform) [5:p.26], which allows for reliable transfers using the inherently unreliable IP-multicast.
3.3 Synchronization of WWW Browsers

When a presentation is distributed over the MBone and a WWW browser is used for presenting the slides, there is a need for synchronization between the involved WWW browsers (this means that all involved browsers display the same page).

This is solved by sending a display-message to all members of the group using the CB (mDesk Control Bus) [5:p.26]. The CB is an agent-based lightweight architecture for simple (but still powerful) messaging within and between CB aware applications. All CB messages are exchanged using reliable IP-multicast.

During the session, all pages that are received are collected in a list. The listener has the choice of either automatically displaying a new page or manually clicking on the list entry to display a new page. If a listener wants to go back and view an already displayed page, s/he can select the page of interest in the list of received pages and that page will be displayed locally. The user can also instruct the local mWeb client to send a display-message to all other listeners including the presenter. This is useful if the listener wants to comment or ask a question related to a page that is not currently displayed.

4 Discussion

The mStar environment can adapt to many education-based scenarios; thus, it is especially well suited for distributed education. In this discussion, we shall put forward our observations gathered while using mStar for these scenarios.

We have noticed that using mStar to teach about its own underlying technology was a very good idea. By doing this, students that take the course are often more technology oriented, and are less hesitant towards using a microphone and video camera to interact. The undergraduate courses at the university are becoming very popular, perhaps because they teach technology using technology.

The statistics from our mMOD server logs show that many students prefer to watch lectures during evenings, or even late at night. Offering the opportunity to study asynchronously has its price; the lectures are becoming less frequently attended. This might not be entirely negative, as courses today are growing in size with sometimes more than 120 students, and it can be very useful for students having overloaded daytime schedules.

Using the playback facilities offers another clear advantage: it enables students to take pauses, to either read additionally related information or to consult the course literature. Unfortunately, these students can not be part of the spontaneous discussions during lectures. Having multiple participants active in the playback environment might remedy this to an extent, but this is clearly an area to be improved.

The multiple gathering of students in groups to listen to the playback of a lecture is also a remedy to the latter problem. This social behavior might come from a need of discussing the material similar to discussions that take place in an ordinary lecture. It might also compensate for the physical isolation brought forth by sitting alone in front of a desktop computer.
We have noticed that other social protocols have been established when using the environment for presentations and education. Foremost are the sub-discussions that take place using the mChat and mWhiteBoard tools, where a set of the participants either discuss the presenters material or something completely uncorrelated. This kind of discussions and sharing of information enhances the learning experience, since attending a lecture physically normally disallows side conversation in the audience.

By encouraging the use of different means of communicating electronically, such as email or WWW-based discussion media, we have found that students tend to help each other. This form of social clustering, a small community within itself, is most interesting. Not all choose to take part, but since a large number does, it lessens the traditional burden of a teacher. Students with additional knowledge have also the opportunity to share it with the rest of the class and the teacher. The fact that students are able to share this knowledge with the group is an enormous advantage to more traditional teaching, where students seems to rarely form groups with more than five members.

An additional observation made using the mStar environment for lectures is that lectures tend to become more static than classical (i.e. non electronic) lectures. Experienced teachers are most often those who can improvise and dynamically alter the course of a lecture. These teachers usually do not need to prepare overhead material, as their lectures often take the shape of a normal conversation. With mStar, teachers are easily 'caught' in the flow of their pre-made electronic material. It is therefore very important to still allow the teacher to improvise, perhaps by adding links to in-depth material from the original presentations and making use of an electronic whiteboard or a sketch board.

Furthermore, a technician is needed to achieve the best transmission quality for the lectures. The technician controls audio levels, camera focus and positions, recording management and lighting in the lecture hall. This means that two persons are needed to conduct a distributed lecture. This extra requirement in human resources should be justified by the fact that no teachers are needed at the 'distance-based' locations. However, the use of movement-tracking cameras and automatic audio level control equipment, can remove the need for the technician.

Traditional distance education methods usually take the shape of TV broadcasts. In comparison with the mStar environment, networked distance education offers more than the ordinary TV broadcasts. Although mStar could certainly be used in a more 'TV-like' environment, such as a one-to-many broadcast media, there are some fundamental differences:

- Both TV and mStar sessions can span long distances, but the main difference is in the setup of sessions. Setting up TV sessions can create many distribution-related headaches. Broadcasting regulations and equipment availability are two major potential pitfalls. With the multicast technology used in mStar, sessions are more lightweight and are easier to create. Multicast sessions also allow for more channels than the two available educational TV broadcast channels in Sweden.
- TV offers no interactivity at all, while net-based education can offer several means for interactivity. Many of these means have been cited in this paper.
Finally, training teachers at remote secondary schools has had a very positive effect. These teachers tend to spread the gained knowledge about this technology and information technology in general, creating a very nice momentum for mStar and for the teachers in general. The fear that knowledge about information technology is decaying at secondary schools in Sweden can therefore clearly be met. For sparsely populated areas like the county of Norrbotten, networked distributed education might be the future. If the Internet is the next industrial revolution, then net-based learning may be the next educational revolution.

5 Summary and Conclusions
This paper describes a novel multimedia environment for distributed education offering many different usage scenarios. The mStar environment consists of a tool suite for preparation of presentations, distributed presentations, playback of recorded and edited multimedia content, and synchronous virtual meetings. These tools and scenarios, such as mWeb, tightly integrate the WWW in a close relationship with IP-multicast technologies.

The variety of usage experiences and the successful countywide deployment clearly demonstrates that mStar is indeed scalable in more ways than one. From small informal presentations to complete university courses, we have shown the strength of this novel education environment.

We have argued this from a variety of perspectives, all showing that this environment offers extended support for interactivity, better help through the use of a 'virtual student community', as well as on-line availability of all course media. The future goal is to create an educational environment that can be qualified as better-than-being-there, bringing normal everyday situations such as interacting, learning and collaboration to the Internet.

6 Future work
The most important future enhancement of the mStar environment is in the field of usability. A survey is ongoing to define metrics and measures about usage in the various projects using mStar. A deeper study could be done by having two user groups, one that follows a course remotely and one that follows it locally, and then compare the results. In addition, better mMOD logs might reveal interesting statistics about usage. These results should help in making mStar easier to use. Using mStar should not be harder than just clicking on a link, especially for primary and secondary school students.

The users of the mStar environment have identified a need for further development, in order to better support distributed education. The most frequently requested functions are:

- An enhanced SlideBurster with support for outline editing, HTML templates using Style Sheets [U19] and incorporation of the new W3C standard SMIL [U20].
- An integrated tool for playback of audio, video and HTML (replacing VIC, mAudio and mWeb). This component should be implemented with the Java Media Framework [U21] to achieve portability.
- A 'pack-and-go' tool could be useful in two ways:
• Packaging of presentations in advance, to support presentations without an Internet connection.
• Distribution and local playback of full recordings.
• Support for a movement-tracking camera together with automatic adjustment of audio volume levels, which will lessen the need for a technician.
• Privacy through encryption of the media, for sensitive or confidential information. This is also needed for 'pay per lecture' education.
• One-to-one audio/video communication within a larger session, for side conversations.
• General application sharing across platforms.
• Remote pointers for pointing certain paragraphs or positions in HTML slides.

Another area of future work is enhancement and expansion of the virtual student community, since spontaneous discussions among students and teachers are vital, even if asynchronous. Adding a shared information space like a WWW based bulletin-board will be investigated, perhaps by using the education framework presented by Lai [4].

References

URLs

[U16] mIndex, <URL: http://www.cdt.luth.se/~nogge/master/>.
[U17] mMOD (multicast Media on Demand), <URL: http://mmod.cdt.luth.se/>.
[U22] Education Direct A project at the Centre for Distance-spanning Technology, <URL: http://www.cdt.luth.se/education.direct/>.

Acknowledgements

Thanks should go to Ulrika Wiss and Johnny Widén at CDT, for interesting comments and nice encouragement. We would also like to thank the project Education Direct [U22], which is supported by the Foundation for Knowledge and Competence Development [U23].

This work was done within Esprit project 20598 MATES [U24], which is supported by the Information technology part of the 4:th Framework Program of the European Union.
Support was also provided by the Centre for Distance-spanning Technology, CDT, and its members.
Paper B

Net-based Learning for the Next Millennium
Net-based Learning for the Next Millenium

Peter Parnes, Kåre Synnes, Johnny Widén, Dick Schefström
Luleå University of Technology / Centre for Distance-spanning Technology
Department of Computer Science
971 87 Luleå, Sweden.

ABSTRACT

Luleå University of Technology has for the last few years deployed a net-based learning environment, mStar, to distribute courses to students independent of time and geographic distance. The mStar environment gives remotely attending students equal possibilities as traditionally attending students to take an active part of a course, as well as enhancing the learning experience for all students. This is made possible through a novel combination of IP-multicast technology and the WWW. This paper report on experiences gained over a few years of practice and depicts a vision of the next generation of the mStar environment.

Keywords: mStar, net-based learning, distributed education, distance education, WWW, Internet, IP-multicast technology.

1. Introduction

If the Internet is the next industrial revolution, then net-based learning may be the next educational revolution [1]. The evidence of this is clearly present at most major universities and companies, where the WWW is used to distribute information to students. Many environments have been presented at past WWW conferences and we can today see that the technology is maturing as usage is increasing.

Some of the early educational uses of the WWW [2,3,4] and virtual classroom environments [5] have been a major influence for this educational revolution. The availability of course related information such as lecture notes, extra course material, exercises, and course schedules blended with the WWW's inherent qualities such as hyperlinks and accessibility have added much information to the classical structure of courses.

A common deficiency found in these early environments are the lack of support for spontaneous interaction between students and teachers. Additional functionality like real-time textual chat and video conferencing has enhanced communication, creating environments that are near to complete in functionality. The mStar environment [1,6,7] is a fully symmetric and distributed system for net-based learning that include the necessary support for spontaneous interaction between students and teachers.

Although the mStar environment is functionally well equipped, the usage of it is still immature. Courses at Luleå University of Technology, LTU, are still given in the
traditional way, with lectures and laborations, and students are not using the available possibilities of interaction.

This paper reports on experience gained from a wide use of the mStar environment for over three years and depicts a vision of the development of net-based learning using this environment in the next millenium. The paper consists of a brief background, a presentation of how the mStar environment is used at LTU today, and finally a discussion on how a future environment could be like.

1.1 Background

One of the major driving forces behind the Centre for Distance-spanning Technology, CDT, is to deploy net-based learning in the county of Norrbotten. The reasons for this is quite evident: a sparse population in a large geographical area, a decrease in funding while students increase in number, a higher demand from industry for post graduate studies, and the increasing problem of finding competent teachers.

Thus, many high schools cannot gather the critical mass, funding and competence to offer the courses and subjects that are possible in the more densely populated areas. By giving multimedia- and WWW-based courses over the networks, a sufficient critical mass is generated, creating a countywide virtual university with breadth and quality that might otherwise not be possible. This virtual university can also be used to guarantee life-long learning aspects, where people continuously can take net-based courses that does not demand their full workday attention. This is becoming more important as people without a university level degree have increasing problems to find work, at the same time as it has become economically harder to complete a full university level education.

LTU have given a large number of distributed courses using the mStar environment, ranging from graduate-courses to full fledged under-graduate courses. The university has achieved a significant deployment and usage of distributed education over the Internet, not only internally but also to other companies and organizations in the county. Giving joint courses might help bridge the gap between local industry and the university.

This paper therefore presents the concrete results of a wide deployment effort of the mStar environment for distributed education where secondary schools, the university, local companies and communities are all active participants. We are now only starting to see the first social and cultural changes within the schools and companies involved. The paper also depicts the vision of the future use and continuation of the net-based learning efforts at LTU.

2. Net-based Learning Using the mStar Environment

The mStar environment is a collection of tools that span from preparation to presentation. The tools are flexible enough to be useable in many educational scenarios, from large scale lectures to small group activities.

Note that by combining the possibilities offered by available networks, the accessibility and ease of use of the WWW and the benefits of IP-multicast [8], we have been able to make
these scenarios a part of our everyday, real-life teaching experiences. We would like to stress that this is a working system in real use.

This chapter means to briefly describe the current use of these tools, as well as some lessons learned.

2.1 Large Scale Distributed Lectures

The traditional large-scale lecture is currently an important part of the net-based learning efforts at LTU; not because it is the best form of education, but perhaps because it is most similar to ordinary education.

To conduct a distributed lecture the lecturer prepares HTML slides before the lecture, where additional content can be linked into the slides for in-depth information. The slides is then put on the WWW in advance of the lecture which give ambitious students the possibility to prepare for the lecture. They can also be used after the lecture for repetition or further in-depth study.

The lecture hall is equipped with computers, cameras, microphones and projectors. This enables projection of the HTML slides in the lecture hall as well as distribution of the slides to remote students attending from the net. In addition audio and video are distributed from the lecture hall, which gives the remote students an equal opportunity to take an active part in the lecture. The remote students can also send audio and video, which gives them the possibility to ask voice questions just as if they were in the lecture hall.

All in all the learning experience come near to something that is 'better than being there'. That is, the remotely attending student has an enhanced learning experience in comparison to the normally attending student. The remote student can in parallel to the lecture follow in-depth material, have side conversations with fellow students, and can even do secondary low attentive. Furthermore, remote students with disabilities have benefits that are not otherwise available, such as control of the slide presentation format (font sizes, colors etc) and a more undisturbed audio environment.

The mStar environment constitutes a set of tools based on IP-multicast. These tools and their respective use have previously been presented in detail [1,6,7], but here is a brief summary of the most used tools:

- mMOD - media-on-demand server for recording and playback of sessions
- Marratech Pro – fully symmetric media client, which consists of several parts:
  - mViewer (slides/sessions, see Figure 1)
  - mVideo (audio/video, see Figure 2)
  - mMembers (corridor, see figure 3)
  - mChat (textual chat, see Figure 4)
  - mWhiteboard (whiteboard, see Figure 6)
The lecturer announces a session for the lecture, then both lecturer and students launches Marratech Pro. All media are then presented locally in the lecture hall as well as distributed to the remotely attending students. The student can ask questions by sending voice (mVideo), text (mChat) or figures (mWhiteboard). Note that the students in the lecture hall can hear and see the remote students and the other way around.
The participants are 'submersed' in a fully symmetric environment that takes distance education a step further from traditional HTML-based courses. The students are no longer passive receivers as they can interact in real-time, which is very important for promoting student participation and debates between class members. Figure 3 shows the mMembers tool (used for the corridor).

![mMembers tool](image)

**Figure 3. Marratech PRO mMembers**

Finally, the lecture can be recorded with mMOD and can then later be played back. A recorded session can for instance be played back into a live lecture, which enable reviewing and debating of related recorded material. The recordings also give students a possibility to rehearse before exams and – more important – follow courses asynchronously. The latter is a major point for life-long-learning issues, where people that cannot attend during day-time or even follow a course full-time actually can take part of a course.

### 2.2 Virtual Student Community

The lectures mainly offers a one-to-many channel, since students often are too shy to ask questions – especially if the session is large. We have found that something complementary is needed, where students can cooperate and interact with each other. It is naturally very important, perhaps even vital, for remote students to have a continuous contact with the teachers and fellow students.

Creating such a community, as described in [5], can also lessen the workload for both students and teachers. Today students themselves are sources of information, the teachers are no more the sole information source, which means that collaboration and sharing of information is vital to make large distributed courses a success.

However, most of the courses currently given using the mStar environment at LTU are ‘normal’ university courses. This means that the incentive to establish a virtual student community is weak, since there is a physical community to fall back on. It is always possible to knock on the teacher’s door. The result is also that the current virtual student communities are sparsely populated. This would change if most students were ‘real’ remote students, and the physical community was not available.
2.2.1 Virtual Teachers Room
The virtual teachers’ room is basically a Marratech Pro session where the teachers of a course are available to answer questions or chair discussions. This could also be a session where a group of teachers are available – thus not necessary linked to a specific group, but as a general resource. The latter case might be important for the feeling of continuity for remote students, since they change courses every 10 weeks and need a fixed point in their studies.

2.2.2 Virtual Group Room
A Marratech Pro session can be used by a group of students, forming a virtual group room, to discuss course related issues, with or without a teacher attending. The virtual group room can also be used for project status presentations, where a subgroup of a larger class is attending.

This form of meetings may prove invaluable, as remote students need to overcome isolation by forming groups and socialize. It is therefore important that courses emphasize collaborative over individual learning, in order to stimulate use of this media.

2.2.3 Virtual Billboards
The value of asynchronous communication should not be underestimated, as many students choose to follow courses entirely or partly asynchronously. Systems that support asynchronous communication are numerous, ranging from simple mailing lists to advanced WWW-based board systems. LTU has recently deployed a system developed internally, W3Cs, which enables students and teachers to communicate via a bulletin board. W3Cs also offer basic support for course management, a task earlier handled by homemade scripts individual to each teacher and course, as well as support for document publication.

One observation made is that usually silent students also contribute to asynchronous communication, maybe because they have more time to formulate their meaning or since the interaction is more abstract. This is more noticeable if a electronic group is maintained over time, creating a virtual community of students where the students ‘feel like home’. The virtual community must however have an active leadership (introducing new members and excluding misbehaving members) and clearly defined boundaries (limiting what should be discussed).

2.3 Lessons Learned
The statistics from our mMOD server logs show that many students prefer to watch lectures during evenings, or even late at night. The possibility to watch recordings is clearly useful for students having overloaded daytime schedules. Using the playback facilities offers another clear advantage; it enables students to take pauses, to either read additionally related information or to consult the course literature. Unfortunately, these students can not be part of the spontaneous discussions during lectures. Having multiple participants active in the playback environment might remedy this to an extent, but this is clearly an area to be improved.
We have noticed that other social protocols have been established when using the environment for presentations and education. Foremost are the sub-discussions that take place using the chat tool, where a set of the participants either discuss the presenters material or something completely uncorrelated. This kind of discussions and sharing of information enhances the learning experience, since attending a lecture physically normally disallows side conversation in the audience. Figure 4 below shows the chat tool in action.

![Chat Tool Image](image-url)

**Figure 4, Marratech PRO mChat**

By encouraging the use of different means of communicating electronically, such as email or WWW-based discussion media, we have found that students tend to help each other. This form of social clustering is most interesting. It lessens the traditional burden of a teacher, where students with additional knowledge often share it with the rest of the class and the teacher. The fact that students are able to share this knowledge with the group is an enormous advantage to more traditional teaching, where students seems to rarely form groups with more than five members.

A downside is that lectures distributed with the mStar environment tend to become more static than classical (i.e. non-electronic) lectures. Experienced teachers are most often those who can improvise and dynamically alter the course of a lecture. These teachers usually do not need to prepare presentation material, as their lectures often take the shape of a normal conversation. With mStar, teachers are easily 'caught' in the flow of their pre-made electronic material. It is therefore very important to still allow the teacher to improvise, perhaps by making use of an electronic whiteboard or a sketch board.

### 3. Net-based Learning for the Next Millenium

The experience gained over the last years allows us to draw conclusions about how a future system for net-based learning could look like. This chapter therefore aims to present our vision and to give initial answers to some of the questions we have found important.

#### 3.1 Questions

Who will the future student be? Students will range from persons reading a single course to persons following a complete fixed program, but the typical student will have a individualized program (where they select course themselves). They may also have varying study paces and learning styles, as well as different needs for learning support. A
conclusion is that the future students will be a very heterogeneous group, especially if we take life-long learning into consideration. This means that courses must be modularized to a larger content than today, so that it is easy to customize a course for an individual depending on the knowledge level of that individual. Required background information and in-depth material should therefore be easily accessible to give individuals an optimal learning experience.

Where and when will the student study? The possibility to study independently of time and geographical location is increasingly important. The idea that a university is for everyone is otherwise hard to attain, at least in Europe. People that are limited to study on evenings or that is resident far from a university are depending on this possibility.

Will the university offer social training? The student used to be a passive individual that is fed information; this is something we need to change. The industry needs persons that can communicate easily and work effectively in groups, which means that we need to better prepare the student for the professional life after graduation. The traditional lectures must be complemented by group discussions and projects where students work in groups. Again this might be a European phenomenon, but is generally important. It is naturally extra important for net-based learning scenarios, where the remotely attending students otherwise easily can get socially isolated.

Will there be a local university? The role of the present universities will not change initially, even if much of the competence will reside on the outside. There are however several efforts to create virtual universities, where top competence is gathered to one virtual location. This means that the physical location for studies will be less important in the future, even if it for obvious reasons always will remain (lab equipment etc). The most extreme is that the university only will be the ‘quality brand’ of an educational program. It might also be so that the currently ongoing competence concentration around cities with universities will slow down, where the less densely populated areas also can take part of higher education and prosper.

3.2 The Vision of the Student Year 2000

Our belief is that students will not be much different than today, but that they will have increased possibilities to take part of university level education. No longer are large geographical distances or time limitations ruling out where and when education can be offered. The student in year 2000 can be everything from a full-time student attending lectures physically on the university, to a part-time student following courses from his home at evenings and weekends. It might even be cheap, both from economical and time perspectives. By giving courses for a larger number of students, they will become cheaper per student. Remotely attending students will also need to travel less, which further lowers the cost as well as the time spent. The student in the next millenium will certainly have demands on cost and time efficient education.

By combining large-scale distributed lectures with group projects and personal assignments, well-balanced distributed courses is attainable. The lectures should contain invited speakers from the industry or other research organizations, to complement the traditional academic content and to better prepare students for what is required after graduation. The group projects leads to early training of social skills, while personal
assignments remain important for grading reasons. However, when courses are attended by a large number of students, it will be increasingly important to establish personal tutors or advisors. Personal relations to teachers are important for the overall learning experience, students simply need someone to ask questions.

The future of net-based learning includes equal parts of freedom and responsibility. We can clearly see that the old way of spoon-feeding students with information is coming to an end, and a more collaborative shared learning experience is forth growing. This however needs students with high self-motivation, discipline and commitment. It also put higher demand on the courses, on content as well as pedagogy. The new net-based courses must apply specially developed content, following the old tracks simply do not work, and new ways of teaching is also required, which emphasize collaborative learning.

3.3 The Future Lecture Hall
The lecture hall will of course continue to play an important role for net-based learning, both for remotely and physically attending students. We believe that net-based lectures will be the 'pulse' in distributed courses, that controls the rate of which the major part of the students will follow the course.

There is however a need to allow more freedom for the lecturer to improvise and break free from the pre-prepared material, such as a slide show. The vision is to combine the already existing tools with an electronic whiteboard, onto which some of the tools are projected and manipulated with a pen by the lecturer. Another device is a tool for navigating the slide show remotely, thus unlocking the lecturer from the position behind the lecturer station and allowing him to move more freely about. This clearly makes the lecture more alive; a lecture where the speaker stays at one position and just turn slides is quite unattractive.

Figure 5 depicts a lecture hall using: A – shared whiteboard with pen support, B – projected virtual corridor, C – projected web slides, and D – lecturer station.

![Figure 5, The Lecture Hall](image-url)
In the setup we foresee the projected on the wall behind a lecturer station, where the lecturer can access tools directly (mainly the chat tool). The whiteboard would be placed in the center, and the slides would be projected on the remaining side.

The whiteboard is shared so everything the lecturer draws on it will also be visible for the remote students, and what the remote students draw will be visible for the lecturer and the students in the lecture hall. The pen-based input mechanism is needed since drawing with a mouse has certain drawbacks: it is hard to draw precise shapes; the lecturer is ‘locked’ to the computer and is not allowed to move around and draw, which gives the lecture a more static impression; and using the pen is more similar to ‘normal’ lectures. Figure 6 shows the whiteboard.

The virtual corridor with the remotely attending students is projected on a screen for giving the physically attending students the feeling of presence. We have previously seen that students tend to be very shy, which might be due to that there are unknown participant and that they feel uncertain by that reason. The visible presence might lessen this effect and yield increased interaction within the lecture hall.

Note that the remote students could be able to be much more interactive than the physically attending students: they can interact using the whiteboard, chat or audio; and they can select to diverge into in-depth material and respond to issues discussed in the main session or side sessions like the chat.
3.4 What about pedagogy?

The current use of the mStar environment yields fairly static lectures, which many students claim are dull and not inspiring. There are a few pedagogical tricks that can be used in order to make the lectures more alive.

One of the simplest tricks is to not stand still! It may sound ridiculous, but it is very effective. The students attending physically will simply have to be attentive in order to follow the lecturer. Using different 'spots' for different modes of a lecture also makes it easier for the student to follow the lecture, knowing when interaction is wished for by the lecturer (basically a social protocol for the lecture hall). Figure 7 depicts a few spots that can be used:

- The chat spot, where the lecturer answers questions from the remote students.
- The lecture spot, where the lecture speaks to the class without many interactions.
- The whiteboard spot, the teacher draws interactively on the whiteboard.
- The discussion spot, where the teacher discusses issues interactively with the class.

![Figure 7, Lecture Spots](image)

This enables a rich social protocol, even if subconscious, in the lecture hall. For the remote students a 'wave' tool can be used, where students that want to ask a question raises his hand by using the wave tool. The lecturer is notified, either graphically or by a discrete sound (or both), and can then choose to answer the question. The question can be asked by using voice or the chat tool, where the latter also could enable moderation.

Another trick is to use more than one media, using at least 3 of the spots, during a lecture. That is, using the browser for presentation of pre-prepared slides, the whiteboard for drawing interactively, and the discussion spot for oral interaction with the class.

Another good way of making the students more active during a lecture is having small exercises. It can be as simple as just asking a question and give the students a few minutes to discuss it in pairs. The remotely attending students can discuss the same question using the chat tool. Then the question is discussed by all to reach a common conclusion.

Lastly, student seminars are good examples on where students are well activated and really have the chance to share knowledge.
3.5 The Digital Library
The library has throughout the centuries been the central institution at most universities, where students have roamed the vast content of recorded information. Today there is a need for a digital library to take the place as the central base for net-based studies.

Several efforts are ongoing for creation of such digital libraries, where not only written information (contained on physical paper or on digital media) is to be stored. The benefits of the digital libraries are obvious:

- Information can be retrieved independent of time and space.
- Everyone have equal access to information.
- Information can be shared or distributed over several digital libraries.
- Information agents can make it easier for the student to select information.

During four years a large number of lectures and seminars have been recorded and placed on a media server at CDT. This example, that is used on a daily basis by students, show that the extension of a digital library is both feasible and useable. We therefore foresee the development of digital libraries containing searchable multimedial content.

This will further release the students of the bond of time and space, enabling net-based and life-long learning not only for a few but for the majority of students. The efforts to modularize and modernize net-based courses, while placing pedagogy in the first rank, make us believe that net-based learning has come to stay.

4. Summary and Conclusions
This paper describes a novel multimedia environment for net-based learning, which tightly integrate the WWW in a close relationship with IP-multicast technologies. The variety of usage experiences and the successful deployment clearly demonstrates that the mStar environment is scalable from small informal presentations to complete university courses.

We have argued this from a variety of perspectives, all showing that this environment offers extended support for interactivity, better help through the use of a 'virtual student community', as well as on-line availability of all course media. The goal is to create an educational environment that can be qualified as better-than-being-there, bringing everyday situations such as interacting, learning and collaboration to the Internet.

5. Future work
Today the different tools are simple and intuitive to use, and the technology is rapidly becoming stable. There is however a long way to perfection, especially considering the pedagogic aspects. Large-scale lectures are still far too common, which in turn are too limited when distributed over the network. Using only HTML slides is too limiting, and using the mouse to draw in the electronic whiteboard is too unprecise. Work is currently proceeding to install a pen-based electronic whiteboard (synchronized with the mViewer Whiteboard) in the lecture hall, to better support spontaneity.
Although the remote students have a very good environment for following lectures, the physically attending students have a less than perfect environment. This is due to bad hardware, where the LCD projector available in LTU’s lecture hall is both too noisy and weak. Together with limited spontaneity, this creates a ’tiresome’ setting for the locally attending students. Hopefully new funding can sponsor a better equipped hall.

As a last issue, the general pedagogic and social issue in our net-based learning environment needs to be studied. How good are distributed courses in comparison to traditional courses, and how does the results of remotely and physically attending students compare? Our belief is that we have a strong technology, but that we are weak on pedagogy. Hopefully, the new Center for Net-based Learning at LTU, CNL, will help us bring clarity to the remaining pedagogical issues.

References


Acknowledgements

This work was sponsored by the Centre for Distance-spanning Technology, CDT, and the European projects MediaSite (IA 1012 UR) and Roxy (Espirit 28410). Thanks should go to Ulrika Wiss at CDT, for interesting comments and nice encouragement. We would also like to thank the project Education Direct, which is supported by the Swedish Foundation for Knowledge and Competence Development.

Marratech Pro is a product from Marratech AB, a spin-off company from LTU/CDT. Many thanks for the heroic efforts made by the people at Marratech.
Paper C

Robust Audio Transport using mAudio
Robust Audio Transport using mAudio

Peter Parnes, Kåre Synnes, Dick Schefström
Luleå University of Technology / Centre for Distance-spanning Technology
Department of Computer Science
971 87 Luleå, Sweden.

ABSTRACT

IP based groupware applications, such as net-based learning environments, rely on robust audio transport for efficient communication between users. This paper therefore gives an overview and an initial evaluation of how to achieve robust transport of real-time audio streams over Internet connections without service guarantees. Due to faulty hardware and network congestion these connections face loss, packet delay and delay variation. Audio streams are especially sensitive due their real-time characteristics, where the end result is degradation of the perceived quality. Packet loss can be repaired using receiver-only, sender-initiated or receiver-initiated techniques. Depending on the actual network condition, an optimal technique can be selected using adaptive behavior together with loss-recovery techniques in the applications. Studies have shown that loss rates up to 20% can be effectively repaired using fairly simple techniques. The paper gives initial results from subjectively evaluating audio quality and presents a research prototype called mAudio that has been used to experiment with different loss recovery techniques.

Keywords: robust, audio, mStar, environment.

1. Introduction

Internet is a rapidly growing phenomenon in many perspectives, perhaps mostly due to the technical momentum created by its development. The first really large use of Internet was for interchange of messages, email, but the large boom in usage coincided with the introduction of the World Wide Web and the sudden instant global access to information. The third large step in the development of the Internet will probably be the introduction of bandwidth demanding media transport with real-time characteristics.

New services include interactive TV, Internet telephony and multi-part conferencing solutions, which all have real-time elements like streamed audio and video. These services require low delay to allow for interactivity, as well as low loss for intelligibility. This paper is focused on how audio transport can be made robust over lossy Internet connections, since transport of real-time audio data over the Internet is particularly affected by delay and loss.

Many modern audio applications therefore include techniques for loss recovery and a few even include adaptive behavior to meet changing network conditions automatically.
However, loss recovery techniques may instead increase delay and bandwidth usage. Tools should therefore be adaptable to the current requirements of a session, where either audio quality or delay is promoted (while keeping bandwidth usage in mind). For instance, a playback of a recording would select audio quality over delay, while a live conference would select the opposite.

In particular, audio traffic transported using the Real-time Transport Protocol, RTP, on the MBone is well prepared for repair algorithms [1,2]. This is due to the presence of a sequence number and a time stamp in the RTP packet header. Several research prototypes like VAT, RAT and FreePhone has shown that real-time audio transport indeed can be conducted with good result, even under lossy network conditions [3,4,5].

This paper gives an overview of networking related to IP Multicast and RTP. It also provides a discussion and comparison of different methods for repairing losses of real-time audio data. These methods are either receiver-only (where the sender is not involved at all), or in cooperation with both sender and receiver. The latter case can be further divided into sender-initiated or receiver-initiated techniques. Observe that many of these methods are not exclusive, and that a combination of these techniques is required to achieve the best possible audio quality. The paper ends with describing a reference implementation of an adaptive audio tool, mAudio, which has been used to evaluate the different recovery techniques described together with a tool for subjective audio quality tests.

1.1 Background
The Centre for Distance-spanning Technology, CDT, at Luleå University of Technology has since the foundation in 1995 conducted research on net-based learning and collaborative teamwork environments. The result is the mStar environment, which is a platform for implementing distributed applications based on IP Multicast [6,7].

Numerous courses have been given using the mStar environment, spanning from small informal graduate courses to full-fledged under-graduate courses with hundreds of participants [8,9,10]. The environment has also been in extensive use within most of the projects conducted at CDT for internal meetings and presentations.

This extensive use has shown that the most important of the different real-time media involved is audio, due to the fact that small disturbances easily can render the audio stream unintelligible. The video has mostly been used for achieving a sense of presence, and the other media are more or less non real-time (chat, whiteboard) since they use a reliable protocol for transport. Efforts have therefore been spent to study how to achieve the best audio quality during different network conditions.

An experimental audio tool, mAudio, has been implemented in order to study different techniques for repair, among them different adaptive algorithms.

2. Networking Issues
While traditional telephony networks are constructed for optimal transport of real-time audio data, Internet is inherently not. Data transported using traditional telephony services will arrive with little delay variation and low loss. The only service offered over Internet,
best effort, give no guarantees on delays or delay variation. This means that Internet applications can neither rely on a guaranteed bit rate, nor assume an uncongested transport service. Several efforts are ongoing to extend the Internet architecture to support more transport services [11,12,13,14]. However, such extensions have not yet been widely deployed. In fact, it may take several years before service guarantees are globally available on the Internet. In lieu of this, IP based applications with real-time constraints should be constructed with delay, delay variation and loss in mind.

Another issue is scalability, where large sessions traditionally have been relying on special replication servers in the network. However, Deering proposed the concept of IP multicast [2], where replication is handled at network level. This alleviates the scalability problems, at least for moderately sized sessions. The IP multicast backbone is referred to as the MBone. The MBone is built with IP routers equipped with software allowing them to forward IP packets not only to a single receiver but also to a group of receivers. These loosely coupled sessions offer clear advantages in scalability over replication-based unicast services, since the amount of traffic sent over the network and the control needed at sender side are minimized.

A sender simply sends its data to the group address, without explicit knowledge about who is receiving. A receiver joins a group by listening to the group address, and is thereby forwarded the traffic by the network. Naturally this level of control is too limited for many real-time applications. The application level protocol RTP is therefore used as a protocol above UDP (Note that RTP is not limited to run over UDP only). RTP also has a control protocol built in, RTCP, which allows the receiver to report back to the sender about its networking characteristics. Each packet distributed with RTP has a timestamp and a sequence number, which makes it suited for transport of real-time data with error recovery in mind.

There have been several attempts to describe the characteristics of the Internet in general and IP multicast in particular, with parameters like loss, delay and delay variation. Due to the heterogeneous nature of the Internet, this has proven to be a hard task. The results are therefore not fully consistent but show common trends regarding IP multicast traffic [15,16,17,18,19,20].

Research indicates that most receivers suffer from loss up to 5% due to faulty hardware and light congestion, while a few experience higher degrees of loss due to greater network congestion. A reasonable design assumption is that if the number of receivers is large, then a packet is likely to be lost at least once by one receiver in the group. The relation between bandwidth usage and experienced loss is clear, which needs to be kept in mind when constructing adaptive schemes for loss recovery.

Unless the network is heavily congested, several consecutive losses are unlikely when packets are sent out evenly spaced [21]. A uniform distribution is therefore a good approximation of losses in networks not suffering from heavy congestion. For more

---

1 Low packet loss is often a result from faulty hardware, such as a badly connected cable or a damaged hub. Minor congestion may also create the same pattern of losses.
detailed simulations, a Markov chain would better describe the loss characteristics of a network, especially when heavy congestion is to be involved.

Lastly, packets that are late due to delays in the network might be dropped by the application. The solution is to let the application have a playout buffer that adapts to changes in delay and delay variation. This allows interactive applications to suffer from a minimal delay for a specified loss rate. Several studies have been made to construct optimal algorithms for adaptive buffers [22,23].

In general, the best way to combat transient losses are to apply techniques for packet repair as described in the next chapter, while long term losses require bandwidth adaptation [24]. The latter will be discussed in chapter four. A possible extension is to also support balancing of reliability and interactability (loss tolerance vs. delay) [24].

3. Techniques for Repair of Audio Streams

This section aims to present the different techniques for creating a loss tolerant audio application. Our focus and experience has been with these techniques applied to IP multicast based tools, but note that they can with equal benefit be applied to IP unicast tools or even regular ISDN applications.

The simplest techniques are those that does not rely on the sender for achieving loss tolerance. These receiver-only techniques are sufficient to cover losses induced faulty hardware and light congestion, that is losses up to 5%. When higher amounts of loss are experienced (typically due to greater network congestion), the sender needs to take measures to lower the losses as well. These repairs are either sender-initiated or receiver-initiated.

Note that the exact percentage of tolerable losses are very subjective and varies from user to user. Also note that more complex repair schemes may increase the delay in the system.

Manipulated sound clips are used in the subsequent subchapters to visualize the effect of some of the techniques described therein. They are all based on a simple phrase, “Hello World”, which is depicted in figure 1 below. Repaired information is colored gray in the following figures.

Figure 1, “Hello World”
3.1 Receiver-only techniques

These techniques are prominent when the recovery techniques that involve the sender have failed to replace a lost packet.

3.1.1 Silence Substitution

This is the simplest loss recovery technique available, where lost packets are replaced by silence. Its simplicity limits its usefulness, as it is only effective up to approximately 1% of loss [25,26]. It may also cause strain, as the clipping effects are quite tiresome.

The packet size affects the effectiveness of this technique, where a packet sizes used for audio usually is either 20, 40, or 60 ms long. A phoneme is about 20ms long and loosing a full phoneme affects intelligibility, thus one lost packet means that 1 to 3 phonemes are lost. For anything but really low losses, silence substitution is therefore bad. Figure 2 shows silence substitution at 40% loss and 20 ms packets.

![Figure 2, Silence Substitution](image)

However, using an additional technique may improve the use of this technique further. An example of this is striping, which is a sender-initiated technique explained later.

3.1.2 Warping

The name of this repair technique, warping, hints that the timing is disrupted during playout. A lost packet is simply ignored and the next in line is used instead, thus resulting in a consumption of the playout buffer when loss occurs. This technique can therefore prove meaningless since the fallback when the playout buffer is consumed is silence substitution while the buffer builds up. It has therefore similar characteristics as silence substitution, but it might complicate the use of advanced adaptive buffers. The biggest downside is however a quite ugly distortion in the flow of the speech. Figure 3 shows the “Hello World” clip using warping.
3.1.3 Noise Substitution

The human brain is equipped with the ability to do subconscious repairs of sound distorted with noise. This is used in noise substitution, where white (or gray) noise is used instead of silence for repair of losses. This is shown to increase the intelligibility as well as perceived quality.

A common way of deciding what noise amplitude should be used is to track the power of the received data, and then base the noise repairs power on this. This however includes the use of silence detection on the receiver side. An alternative is that the sender uses silence suppression, only sending audio when necessary, or only use silence detection to calculate a noise power that is then sent to the receiver out-of-band.

This technique is slightly better than silence substitution, and thus gives a higher loss tolerance. Note that the selection of the noise waveform is important, where selecting a too high noise may actually lessen the subjective gain in audio quality. Figure 4 shows noise substitution repairs based on the mean amplitude of previous packets.

An alternative way would be to add a continuous noise to the signal, that act as background noise. The power of the noise should then be calculated on the loss rate and the average power of the real signal. In GSM, the term for this is comfort noise. It is also possible to only add the continuous noise when needed, that is after a certain degree of loss, in order to further minimize the disturbance caused by the overlaid noise.
3.1.4 Repetition

A technique introduced by GSM [27] is to use previously received data for repair. Simply put, take the last packet and repeat it if the current packet is lost. GSM uses subsequent repetitions for as long as 320ms, when using a data size of 20ms (or 33 bytes). The repeated packet is slowly faded until silence.

This works quite well, since speech waveforms often exhibit a degree of self-similarity. That is, nearby located packets will show similar spectral qualities. As a result, repetition works well up to 5-10% of loss.

An advantage with repetition is that it is quite simple to implement. The disadvantage is that quite ugly reverberation effects can occur if repetition is overdone. A recommendation would be to keep the repetition small, which works fine for moderate loss due faulty hardware or low congestion. Using 40 ms packets a repetition scheme of two subsequent repetitions with an amplitude gain shift of 50% each works quite well, while if using 20 ms packets three repetitions with an amplitude gain shift of 33% is recommended. Figure 5 shows repetition of 2 packets at 50% gain shift.

![Figure 5. Repetition](image)

3.1.5 Forward repetition

The notion of repairing a lost packet with a subsequent packet is similar to the normal repetition, but work only for small number of subsequent losses.

A combination of normal and forward repetition may be the best solution, especially when subsequent losses occur. Using the latter combined technique increases the loss tolerance above the simple repetition technique.

3.1.5 Mixing

Mixing the surrounding packets and applying an amplitude gain shift is another technique that works well for low losses. This keeps much of the spectral qualities of the lost packet, especially if using a 20 ms or smaller packet size.

If more than one packet is subsequently lost, the surrounded packets can first be gain manipulated. This in order to find the best balance when mixing the two packets, and thus achieve an as accurate as possible repair. Figure 6 shows mixing with distance balancing (where the amplitudes of the two surrounding packets are amplified depending on their respective distance to the lost packet).
3.1.6 Interpolation
This technique is based on studying the spectral qualities of the packets surrounding the loss. Goodman et al [28] has studied interpolation limited to preceding data as well as surrounding data. This technique may be computationally demanding and does not give much better results than mixing.

Note that mixing can be seen as a simple form of interpolation, where more complex interpolation techniques uses the spectral qualities of the surrounding packets in order to achieve a more accurate repair than is possible when mixing.

3.1.7 Stretching
A lost packet can also be repaired by stretching the surrounding packets to cover the loss. This is like interpolation computationally demanding, but performs a little better. A downside may be spectral effects that are introduced when manipulating the samples, similar to warping.

3.1.8 State Interpolation
The GSM speech coder [29] is an example of an encoder that uses linear prediction. This makes it possible to interpolate the linear predictor state, and thus achieve a repair based on knowledge about the speech encoder. This is however a complex and very computationally demanding process and the much simpler mixing technique achieves similar results.

3.2 Receiver-initiated techniques
By receiver-initiated we mean that the sender is the active party for preventing loss. Observe that most of these techniques use redundant data, which may not be used at all at the receiver side and thus adds bandwidth. This may render these techniques useless under the condition of network congestion.

The optimal solution is to combine receiver-initiated techniques with bandwidth adaptation methods, as described in chapter four.
3.2.1 Striping
Striping means that data from several packets are interchanged. A single loss therefore incurs several small losses instead of one large. An example if we use a 5 packet striping technique using 20 ms packets, where not one 20 ms will be the effect but 5 4 ms losses. This greatly increases the efficiency of most receiver-only techniques.

One downside is that a computational overhead for moving the data around, but the gain may be greater when using striping together with for instance silence suppression than using a more complex technique like state interpolation. Another downside is increased delay. Figure 7 shows striping using a 3 packet striping technique.

![Figure 7, Striping](image)

3.2.2 Interleaving
Interleaving is very similar to striping, but interchange hole packets. It is therefore not as computationally demanding, but does neither give as good results. The method is however quite effective for small packet sizes (~20 ms), and can have the good effect of reducing subsequent losses due to congestion. Assume that we have a sequence of packets A…F…, transmitting them like A-C-B-D… or A-D-B-E-C-F… will avoid two subsequent packets to be sent immediately after each other.

The downside is naturally increased delay, but as with striping no extra data is sent which is good for the overall bandwidth consumption.

3.2.3 Media Independent Forward Error Correction
Recent research have reused the ideas behind algebraic codes designed to detect and correct errors in a data stream to generate redundant data to increase the ability to recover from loss. Rosenberg et al has proposed parity coding and Reed-Solomon coding for inclusion in the RTP payload specification [30].

The idea is quite simple, a repair packet is generated of n packets and this repair packet can be used together with n-1 packets to regenerate a single lost packet (of the n packets sent).

The biggest advantage with this technique is that is media independent, but it is also one if its biggest disadvantages. In the case of audio data, information about the data can increase the usefulness of the FEC coding. It is however fairly trivial to implement media
independent FEC, given that existing coders can be reused. Another disadvantage is that it like all FEC techniques increases the bandwidth requirements.

3.2.4 Media dependent FEC

Hardman et al [4] suggests transmitting audio data multiple times, and thus can use the redundancy to repair losses. This technique is very simple, but increases the bandwidth requirements like the media independent FEC. Podolsky et al [31] and Bolot et al [32] have also studied this type of FEC.

Depending on the bandwidth restrictions, several different combinations of main and redundant data can be used. For instance, the main encoding can be PCM while the redundant data is encoded using GSM to decrease the bandwidth requirement. Another example is to use GSM in combination with LPC instead. The more synthetic GSM and LPC coders is well able to cover for small losses, since they preserve the frequency spectra (LPC is consider to contain ~60% of the speech information, while preserving most of the frequency spectra).

Redundant information is usually encoded using a low bit rate encoding, since these are fully capable of covering small losses. These could also be selected adaptively, to face changing network conditions, where even multiple redundancy can be used if the net suffers from high levels of loss due to faulty hardware while being otherwise uncongested.

The redundant information could be transmitted in several ways. First it can be sent as extra packets in the same data stream (or multicast group) as the main encoding. Secondly it can be piggybacked to the same packets as the main encoding, thus decreasing the overhead for headers and parsing. Lastly it can be sent in parallel as separate data streams (or multicast groups), which enable the receivers to select if they need the repairs or not. The last can be good when facing receivers with inhomogeneous networking conditions, where adding redundancy in some cases will increase congestion greatly.

An advantage is that the redundant data could either be sent immediately (if not piggybacked), or be delayed one to several packets. This allows for a sender to trade delay against increased loss tolerance (and thus better audio quality).

Note also that a receiver could select to ignore the redundant data in order to achieve a lower delay. This allows for a differentiation of the receivers, where a real-time conversation would require low delay while a recording would emphasize a high loss tolerance.

3.2.5 Simple Layering

By sub-sampling the main high-quality audio source, it is possible to achieve layered encodings. A simple example of this is to use a 32 kHz audio stream, then sub-sample and filter it to 8, 16 and 24 kHz. The receiver can then select to join the best possible stream (multicast group), depending on the current network conditions, as well as joining several streams to achieve a higher loss tolerance where a lower quality stream is upsampled and used as repairs.
Note that this can be used in combination with FEC, where higher qualities is sent directly and lower qualities uses the FEC techniques. The receiver can then better adapt to networking conditions and select the best use of the available bandwidth.

### 3.2.6 Wavelets
Another form of layering is the use of wavelets, which is a spectral encoding. Wavelets make it possible to separate a high quality audio stream into frequency bands. For instance, a 32 kHz stream could be separated into 0-8, 8-16 and 16-32 kHz subbands. Each subband is the transmitted on its own stream (or multicast channel) and the receivers can simply select the best possible configuration according to the current networking conditions. The received subbands are then added and played back.

The big disadvantage of using wavelets is that it adds much delay, and thus is not suitable for interactive applications. It is also quite computationally demanding, even if it achieves a good level of compression.

### 3.3 Receiver-initiated techniques
In order to avoid sending redundant data that is not used anyhow, a basic idea would be to let the receivers tell the senders when loss occurred. This is a good idea, but for obvious reasons it is hard to find a useful scenario of usage for this kind of techniques. The main problems are delay and scalability.

#### 3.3.1 Reliable Transmission
One technique for reliable transmission is the use of Scalable Reliable Multicast, SRM [33], techniques. When a receiver detects loss, it will request a repair from the sender. This is done after a random time depending on the number of receivers and the distance from the receiver. If other receivers loose the same packet, they will suppress their request if they see another receiver’s request for that same packet. Any receiver that intercepts a repair request may send a repair if available.

The huge downside to this technique is delay, since the repair can be delayed for a relatively long time. As a result, this technique is not suited for interactive applications. The worst case is also that at least one receiver loose each packet, which is possible for a large number of receivers, where the network bandwidth requirement is doubled. Naturally, the retransmitted repairs do not have to be of the same quality as the main transmission.

#### 3.3.2 Semi-reliable Transmission
The idea with semi-reliable transmission is to time limit the repairs, thus trading loss tolerance for lesser delay. That is, if a repair has not arrived within the limited time, then ignore it and repair it with other means.

However simple, this makes it possible to differentiate the receivers much like with the FEC techniques. The usefulness is limited though, as a FEC technique is simpler for much the same effect.
4. Adaptive Applications

Due to the varying conditions of the network, applications need to adapt in order to achieve the best possible result. The need for trading delay versus loss tolerance, and bandwidth required versus loss tolerance are but two adaptations that can be made automatically.

To do so, mechanisms for detecting loss due to faulty hardware and congestion is needed. Then these mechanisms should be used to avoid congestion yet achieve the best possible perceived quality.

4.1 Network Metrics

The use of RTP allows the receivers to make several measurements on the network conditions. First, loss rates can be calculated using the sequence number. These can be calculated with short term or long term loss in mind (due to faulty hardware or congestion), as well as in comparison with other streams in order to decide if the loss is local or distant.

Since audio data is timed, it is also possible to use that timing information in order to decide the delay variation. So all necessary metrics are available in order to be able to adapt to changing network conditions.

4.2 Congestion control

As described in chapter three, there are many techniques for adapting to the existing bandwidth. No standard protocol for controlling the congestion exist however, even if some are suggested [34,35].

A good start is to use a combination of media dependent FEC and receiver-only techniques, together with layered encodings based on multicast group separation. These combinations will be able to remedy most cases of loss, where the receivers play a central role for their own error tolerance. It is by that also possible to adapt to the role of the receiver, be it a telephony application or a session recorder.

Several efforts have been made to study an optimal control mechanism for real-time traffic [36,37,38,39].

4.3 mManager

[40,41] presents the architecture and implementation of a new proposed framework for control and management of software applications. It allows applications to distribute messages in a scalable way, with regard to both the numbers of applications currently running and the available control bandwidth. This is done using IP-multicast together with a messaging platform called the Control Bus (CB) and a reliable multicast protocol (SRRTP). Note, that the whole framework is designed without any special requirements from the underlying transport mechanism as long as it is transport reliable and uses IP-multicast (unicast can be used but the framework becomes much less scalable then).
The novel usage of IP-multicast for management and control creates a mobile and scalable framework that can be used for a number of different applications including better service for distributed audio applications.

The management framework could be used within distributed real-time media applications to signal out-of-band cues to give better relative service within a session to important media senders. For instance, when a user is sending audio, video transmitters within a session should lower their targeted bandwidth. Another usage scenario is to allocate bandwidth between users and sessions depending on external input. For instance, important sessions should get more bandwidth allocated to them than less important sessions. This leads to a system where media applications cooperate both within and between sessions instead of competing for bandwidth, as is the case in today’s global media sessions.

5. Subjective Evaluations of Audio Quality

One way to decide the usefulness of loss recovery techniques is to make subjective tests of the perceived audio quality. The effect of delay is already well described where a delay of more than 250 ms are generally considered to affect an interactive application negatively.

For this reason, a test bed has been implemented in order to do subjective trials of perceived audio quality with different loss recovery techniques at varying loss rates. The initial results confirm the results of other subjective tests [4,42]. However, the initial evaluations done by the authors may not be statistically assured, since too few users have been tested so far.

The most commonly used technique to subjectively evaluate audio quality is to use a full quality sample and a degraded sample, where the user then rates the degraded sample from 1 (equal to the full quality sample) to 5 (significantly distorted). If you combine this with some method for equalizing the variations among the users, then it gives a fairly good measure of the overall perceived quality. The downside is that the lower end of the scale (to decide what are significant distortions) is very personal and thereby varies a lot.

In our subjective evaluation of audio quality we therefore used three samples instead; a full quality sample, a degraded sample and a distorted sample. The user instead grades the degraded sample between the full and the distorted sample, and does so by freely replaying any of the 3 samples before deciding. This removes the uncertainty of the lower level of former method, and also lessens the need for equalizing the variations among the users.

Each user typically graded 50 degraded samples of speech, which were randomly selected and manipulated with loss and recovery methods. The loss rates used were 2, 5, 10, 15, 20, 30 and 40%. The pilot recovery methods used for this initial evaluation were silence substitution, noise substitution, single redundancy, single repetition and double repetition. These were selected as a starting point for evaluation of more advanced schemes.

The evaluation was conducted by 37 users, of which 65% were male and 85% were used to talk in mobile telephone. Figure 8 shows the user interface of the test application, where the user is asked to subjectively grade degraded samples between “good” quality (the original
full quality samples) and “bad” quality (the sample distorted with 40% loss repaired with silence substitution)².

The metric perceived quality is ranged from 0 to 100, where 0 and 100 is equal to distorted and full quality respectively. An important question is where the level of acceptable degradation is located on this scale. This was found to be between 2 and 5 % loss, above which the distortion was clearly disturbing.

The evaluation confirms that noise substitution is slightly better than plain silence substitution, in general. However, many of the users reacted on the noise as quite disturbing, which may be an effect of the actual noise selected. The noise is based on a randomization using a uniform distribution, without any filtering for higher frequencies. As a conclusion, either a prerecorded noise or a generated noise with filtering will be used for further studies. Another conclusion was that a continuous noise signal based on signal power and loss rate would increase the effect of noise substitution further, as the short bursts of noise is quite disturbing. A more uniform presence of noise would be preferable. See figure 9 for a comparison between silence and noise substitution.

² This conforms to the Good and Bad buttons in the tool.
Figure 9, Silence vs. Noise Substitution

Using redundancy increases loss tolerance greatly as expected, while repetition yielded an unexpected result. The double repetition technique was considered slightly worse than the single repetition. Asking the users proofed that they perceived the audio as metallic, with ‘tin can’ effects. Figure 10 shows the effect of redundancy together with silence substitution and figure 11 shows single vs. double repetition.

Figure 10, Redundancy

A final conclusion is that perceived quality for even moderately lossy (around 20%) audio streams can be achieved when using simple techniques as repetition and noise substitution together with redundancy. The perceived quality for this combination reaches the level of acceptable degradation.
6. **mAUDIO**

The mAUDIO application is a VAT/RAT compatible research application, which has been used to evaluate different loss recovery techniques. mAUDIO has been used as an integrated component in the mDesk application for conferencing and net-based learning, and has also been used for recoding and traffic concentration in the mTunnel tool [7,43]. It is implemented in Java and C for Solaris and Windows. Figure 12 shows an early prototype of mAUDIO for Solaris from 1997.

![mAudio Prototype](image)

**Figure 12, The mAUDIO Prototype**

mAUDIO uses adaptive playout buffers together with a fairly simple loss recovery algorithm (the annex describes the algorithm used as pseudo code for 40 ms packets). The algorithm is based on packet repetition, together with noise substitution. The play-out buffer size is adjusted by tracking the sequence number of the RTP packets. This implementation is
simple, but quite effective for small losses. More complex techniques, for instance mixing, striping and parallel FEC techniques were also implemented.

The block diagram of the data paths is also quite simple, and is depicted below in figure 13. The device captures audio, which is forwarded to a silence detection algorithm (based on the mean amplitude) and to the mixer for loopback. The encoder receives audio not below the silence detection level, encodes it (currently only GSM and PCM is supported) and feeds the transmitter with the encoded audio. The transmitter sends the audio data to the network using RTP. The receiver divides the traffic into several streams, and puts the data into one adaptive buffer per sender. The mixer requests data from a number of byte buffers (again one per sender), which fetches and decodes data from the adaptive buffer upon need.

![Figure 13, mAudio Block Diagram](image)

The usage of Java showed to be a limitation however, as the Solaris implementation turned out to have severe resource problems with heavy real-time traffic. This resulted in high CPU usage (typically 20% on a SUN UltraSparc 170E using JDK 1.1 with green threads), together with severe timing problems on lesser SUN workstations (such as the Sparc5). The application was therefore kept as simple as possible, to reduce CPU usage. Recent just-in-time compilers with native thread handling have made it much better, basically lowering the CPU consumption with 30% while removing most of the timing problems on slower systems. Generally, the play-out is quite robust, while the recording is more sensitive. This is due to that the device itself also keeps a play-out buffer, while the more direct recording often lead to long delays when the machine hangs during thread switching. These tendencies are however in common with the Windows platforms, especially on the Windows 95 side. A larger play-out buffer can remedy this, but it also creates a lot of additional delay (the Windows Win32 wave device may stall for 500 ms in rare cases during capture). Using really small clips (20 ms or less) improves the smoothness of the capturing under Windows, but it takes more CPU.

The mAudio prototype is today developed further to a building block of the commercially available conferencing tool Marratech PRO. Figure 14 shows the mAudio component,
integrated with a video component in Marratech\textsuperscript{3} PRO mVideo, used in a net-based learning setting at Luleå University of Luleå.

![Marratech PRO mVideo](image)

\textit{Figure 14, Marratech PRO mVideo}

### 7. Conclusions

This paper gives an overview of available techniques for loss recovery for audio streams. It shows that a great deal can be done to achieve fairly good loss tolerance using simple techniques. The paper also points at the trade-off between loss tolerance and delay, and on the point that an application should be implemented to adapt to it’s networking conditions and use.

Initial results of a subjective audio quality evaluation are also presented together with an experimental prototype for audio conferencing. The results confirm previously done research, which shows that up to 20\% loss can be tolerated when simple recovery techniques are used. One algorithm, used in the mAudio tool, is also presented that give a

\textsuperscript{3} Marratech AB is a spin-off company from the Centre for Distance-spanning Technology founded in 1998 to commercialize the ideas of multicast-based conferencing tools.
reasonably good loss tolerance under network conditions with faulty hardware and light congestion.

Creating IP-based groupware applications, such as net-based learning environments, that are resilient to loss, delay and delay variation is therefore possible.

7.1 Future Work

Clearly the work on the subjective audio quality evaluation needs to be expanded to achieve greater statistical precision, together with the addition of more complicated loss recovery techniques like media dependent FEC. The study also focuses on perceived audio quality alone, and thus give little information about intelligibility.

References

Paper C - Robust Audio Transport using mAudio


Acknowledgements

This work was sponsored by the Centre for Distance-spanning Technology, CDT, and the European projects MediaSite (IA 1012 UR) and Roxy (Esprit 28410). Thanks should go to Ulrika Wiss and Johnny Widén at CDT, for interesting comments and nice encouragement. We would also like to thank the project Education Direct, which is supported by the Swedish Foundation for Knowledge and Competence Development.

Marratech Pro is a product from Marratech AB, a spin-off company from LTU/CDT. Many thanks for the heroic efforts made by the people at Marratech.

Annex

Pseudo code for the mAudio repair algorithm for 40 ms packets:

```c
int cnt = 0; // Number of consecutive lost packets

byte[] read() {
    if (received(n)) { // main or redundant packet
        decreaseBuffer(); // adaptive buffering
        cnt=0;
        return decode(n);
    }
    increaseBuffer(); // adaptive buffering
    cnt++;

    if (cnt == 1) // Repeat with 50% amplitude
        return amplify(n-1, 0.5);
    if (cnt == 2) // Repeat with 25% amplitude
        return amplify(n-2, 0.25);
    if (cnt < 10) // Feed noise with correct amplitude
        return noise(n-cnt);
    return silence; // Feed silence
}
```
Paper D

The CDT mStar environment: Scalable Distributed Teamwork in action
The CDT mStar Environment: 
Scalable Distributed Teamwork in Action

Peter Parnes, Kåre Synnes, Dick Schefström  
Luleå University of Technology / Centre for Distance-spanning Technology  
Department of Computer Science  
971 87 Luleå, Sweden.

ABSTRACT

This paper presents the mStar environment, which creates an environment for truly scalable 
distributed teamwork. It can be and is being used on a daily basis for electronic meetings, 
distributed electronic education and daily work. It creates a new teamwork environment, 
which allows users to collaborate even if they are not present at the same physical location.

The mStar environment includes: the multicast WhiteBoard - mWB, which allows for 
collaborative reviewing of text and images; mChat, which allows for text based group chat; 
mVote, which allows for distributed voting and mWeb for shared WWW objects. These are 
all desktop and IP-multicast based and symmetric.

The mStar environment also includes mMOD, which is a VCR-like tool for recording and 
playback of teamwork sessions, and mTunnel, which is an application for handling IP-
multicast traffic on narrow links in the network (such as ISDN/modem) and network 
segments that does not support IP-multicast. It allows for scaling and transforming of the 
network based data in various ways.

Keywords: IP-multicast, desktop conferencing distributed presentations, digital recoding, teamwork, 
distance education, better-than-being-there.

1. Introduction

This paper presents the mStar environment for scalable distributed teamwork, and how 
desktop audio/video conferencing combined with a number of useful tools can be used to 
create a distributed teamwork environment for daily use.

1.1 Background

The Centre for Distance-spanning Technology/CDT is involved in several different types of 
projects, where 'types' mean where members are physically located, which working-hours 
they prefer and how they tend to work. Most employees at CDT have offices in the same 
corridor and work full time there, but some persons only work part-time at the CDT and 
often switch between different offices. There are even persons that do not work in the town 
where CDT is located. Others prefer to work most of their time from their homes. Further,
there are some employees that prefer to start working early in the morning (even before 6am) and others that prefer to arrive around lunchtime.

All these different working aspects make it harder for some persons to collaborate and interact, and much of CDT’s work has been focused on this distributed teamwork issue. As an answer to this need for a true distributed environment, the mStar environment was developed. mStar creates a scalable distributed teamwork environment that can be and is being used on a daily basis.

The need for mStar is also well documented as part of the rationale for CDT itself. It has for a long time been evident that almost no project of any significance is located in a single place. Physical distribution of participants is a just a fact, and mStar is directly targeting this problem, potentially turning the distribution into an advantage. The mStar development project was initiated with the ambitious slogan of creating the better-than-being-there project environment - a goal that seems clearly in reach.

In a longer perspective, the generality of the underlying technology, which may be applicable in many contexts, ranging from informal meetings, large presentations, education and even media and variations on the idea of interactive TV, is essential for the project itself.

1.2 Scalable Distributed Teamwork

There exist a number of reasons for groups of people to communicate using computers instead of meeting face to face, for instance the cost in both money and time to travel to and from meetings. Another use for mStar is as a creator of group awareness between project and department members which are not located at the same physical place, or which spend part of their time in different offices.

A number of different technical and social aspects of distributed teamwork exist, of which the research presented in this paper primarily concentrates on the technical aspects of real-time synchronous and asynchronous teamwork.

1.3 Organization

The rest of this paper is divided into the following sections:

- Design Issues, about the different design issues selected for the development of the mStar environment.
- The mStar Environment, about the mStar itself and its components.
- Developing New Teamwork Tools, about how the technology underlying mStar can be used to create new distributed teamwork applications.
- mStar Usage Scenarios, about how mStar can and is currently being used.
- The Education Direct Project, about the mStar environment has been used for distance education outside the University and the research environment, involving “normal” non-technologists.
2. Design Issues

Three major design-issues have been dominant in the design of the mStar teamwork environment: scalability and robustness through IP-multicasting access from the desktop and symmetry in the applications.

2.1 Scalability and Robustness

Most existing teamwork and group environments are based on a client-server architecture, with a central server using unicast between the client and the server. When the number of users or the amount of traffic increases, this central server often becomes the bottle-neck. The server also becomes a single point of possible failure as every client depends on the server. The use of unicast makes the systems less scalable, as all traffic between members of a teamwork session has to pass through the central server and traffic on shared links will be duplicated, even if it carries the same data at the same time.

In the design of the mStar environment, the use of IP-multicasting [1] has been a key issue. To create a fully distributed environment there is no central server. The use of IP-multicasting means that traffic between members is only duplicated where needed in the network. This does not imply that each member has to keep track of every other member that needs a copy of the sent data, but instead this information is stored and updated by the network and its routers. This of course also means that the traffic between members is sent on the local network and over intra- and internets and does not rely on dedicated ISDN connections, as is the case in H.320 (ITU-TS published standard for ISDN based audio/video and multi-party conferencing) based systems.

The usage of IP-multicasting also leads to a larger degree of robustness, as users are not dependent on a single central server. The usage of the IP-multicast based tools may continue even if a failure occurs to the server or the network between the user and the server.

The tools using IP-multicasting on the Internet are loosely called the MBone applications. The name MBone is also used for the virtual and experimental network¹ for distribution of IP-multicast traffic on the Internet.

¹ This virtual network is being integrated into production network.
2.2 Access
In many teamwork environments, real-time group communication is based on shared equipment and specially dedicated rooms. This means that distributed electronic meetings have to be scheduled in advance and participants have to move to the dedicated room. This often becomes the bottleneck in the teamwork environment, as these rooms tend to be used exactly when You need them and have to be booked far in advance, which prevents spontaneous electronic real-time communication.

In the mStar environment development, one key issue has instead been that all tools should be accessible from the desktops of the involved users. For instance, a project member should be able to call another project member using the computer on his desktop and be able to use it for real-time audio/video conferencing, shared whiteboards, etc., together with the other party. This means of course that each desktop computer has to be equipped with a frame-grabber and a camera, but this is no big problem as the cost of these products is already very low and falling rapidly.

An advantage of having the necessary equipment on the desktop is that it can be used all time and in new and different ways. For instance, the users can have a "conference" running 24 hours a day and by that creating better group awareness. A new usage pattern is evolving, which resembles electronic corridors more than specific meetings, where users can and do meet spontaneously to talk about anything they want.

2.3 Symmetry
A serious problem with many client/server real-time systems is that they are designed with a “hard” client/server architecture, meaning that clients are designed to only receive data and not to be able to transmit any data themselves to other members. This makes the systems useful for broadcast of real-time data, but provides no functionality for communication with other listeners.

All mStar tools have been designed with symmetry in mind, meaning that any member of a session can transmit just as much as any other member.

3. The mStar Environment
The mStar environment consists of a number of loosely coupled tools that together create a scalable distributed teamwork environment. The basic components needed in this distributed teamwork environment are desktop audio and video, other components needed are shared whiteboard, chat, voting, and distributed presentations. The mStar environment also contains a library for developing new teamwork tools. This library, its contents and architecture, is further discussed in Section 3.

This section describes the different tools that are part of the mStar environment.

3.1 Session Directory Tools
Each conference is referred to as a session and may include any type of media. To allow users to find each specific session they want to join, announcements about active and future
sessions are multicasted to predefined multicast groups. Each announcement contains meta-
information about a session, such as media, contact information, pointers to more
information, when the session is active, and the network addresses to use.

Sessions can either be public, or private, where the first can be compared to TV broadcasts
where the user tunes into the right frequency when switching channels (with the difference
that on the MBone, the user can be a member of several sessions at the same time).
Information about private sessions is not publicly announced, but is instead propagated by
some other method such as email or multicasted encrypted.

A user may be a member of any number of sessions simultaneously and as there can be
several sessions announced at the same time, tools to visualize this to the user are
necessary. These tools are called session directories. Within the mStar environment, there
are currently two different tools available for viewing information about current sessions,
the Session Directory – SDR [2] and the multicast Session Directory – mSD [3].

SDR is a stand-alone tool that displays information about currently announced sessions and
can be used to launch tools corresponding to the media used within a session. It can also be
used to announce new sessions.

mSD is a similar tool to SDR, but instead of being a stand-alone application it uses the
World Wide Web as its user interface. The advantage is that every user do not need to run a
copy the directory tool itself, instead one copy is enough per local server as all users can
access the same user interface. This of course makes it client/server based but as it includes
its own minimal WWW server it is very easy for users to start their own copy. mSD relies
on the mLaunch application [4], which is a World Wide Web browser helper application for
automatic startup of the necessary media tools.

Note that mSD can not announce sessions itself, instead that is delegated to another
application called mAnnouncer [5], which runs as daemon in the background announcing
sessions. This has the advantage over SDR, in that SDR must be running to make the
announcements, as that is not handled by the network but by the application itself. It means
that the user have to be logged in and have the tool running during the announcement
period.

mSD, mLaunch and mAnnouncer are all developed at CDT.

3.2 Audio Tools

The group environment must include a tool that allows users to talk with each other using
their desktop computers. There are a number of MBone tools available for this, including
Vat [6], Robust Audio Tool/RAT [7] and Free Phone/FPhone [8]. All these tools are
compatible, but the newer RAT and FPhone also include support for redundant encodings
(which provide better sound over congested links).

Within the mStar environment, Vat is primarily used (but certainly not a requirement). It
allows a group of users to communicate using speech and sound, using their desktop
computers. Within a session a user can talk to all other members of the session, which can
be compared to talking loudly in an office landscape where everybody can hear what is
being said. This is useful if the user wants to ask the whole group a question or just announce something. A user can also start a private dialogue with another user, which can only be heard by these two users and not the rest of the group. If no one says anything, no data will be sent on the network, which allows users to be members of several different sessions without wasting bandwidth.

A tool called the multicast Radio, mRadio have been created at CDT for experimenting with audio of higher quality, such as CD quality and new mechanisms for handling packet loss, such as semi-reliable transmission where applications are allowed to request lost packets from other listening members of the session.

3.3 Video Tools

The video component of the mStar environment allows users to transmit a video view from their workspace using an external camera. This is done by using the Vic tool [9], which allows users to both transmit as well as receive networked video. The video component adds a feeling of virtual presence where users can peek into other users rooms, both to see what they are doing but also if they are physically there. See figure 1 and 2 for snap-shots of the Vic user interface and what a transmitted view might look like.

![Figure 1, The Vic application. Overview over all members currently transmitting in a session.](image)

The video encoding used allows for sender decided quality, which primarily depends on the available bandwidth. To keep down the bandwidth usage, only low quality video is usually transmitted. The video encoding also allows for dynamic frame rate depending on how much is happening within the video-source, meaning that if nothing happens almost no data will be sent.

---

2 Some low-bandwidth background control traffic will still be transmitted.
3.4 The multicast Desktop - mDesk

The multicast Desktop - mDesk, is a teamwork application currently consisting of three functionally different tools. The different tools are started from a common toolbar (see figure 3).

mDesk can very easily be configured to contain new tools. mDesk has been developed at CDT. The architecture and design of the mDesk framework and application is further discussed in [10].

3.4.1 mVote

Within every official meeting there is often a need to be able to vote about some issue. This functionality is accomplished by the mVote tool. mVote allows users to create new issues, vote about these issues and view a summary of the voting. When creating new issues, users are not limited to the normal yes/no/abstain alternatives, but can create alternatives of their
own. The result is presented in real-time (when members vote) as numbers and in graphical format (as bar- or pie charts). Figure 4 displays a snapshot of the mVote user interface.

![Figure 4, The mVote application.](image)

### 3.4.2 mChat

In some cases the audio is not available. It may be hardware problems or that someone else is talking and should not be interrupted. Some information is, finally, just better communicated in written form. The mChat tool allows users to textually chat with each other. mChat has shown to be especially popular during meetings when users are “forced” to listen to someone else. Figure 5 displays a snapshot of the mChat in use.

![Figure 5, The mChat application.](image)

### 3.4.3 mWB

mWB is a tool that allows users to share a common electronic canvas where they can draw together, share images and text-documents. mWB can be viewed as a digital representation of a physical whiteboard. This digital representation has a number of advantages over the physical, such as several pages, loading and saving of drawn data, so called mini-views that display a small view of other not currently displayed pages, different colors and fonts,
moving/copying of drawn objects, layers and remote pointers. Figure 6 displays a snapshot of the mWB work area.

Figure 6, The mWB work area.

3.5 The multicast Web - mWeb
Within any distributed teamwork environment there is always a need for doing distributed presentations, and within mStar the presentation media chosen is the World Wide Web. There are many advantages of using WWW as the presentation media, including:

- The visual quality of the slides get much higher as users may decide themselves, how the slides are to be presented. Also, in many distributed presentation environments, slides are filmed using a camera from a projection screen, and thus resulting in both much lower quality and much higher bandwidth requirement.
- The usage of WWW makes the slides (they are normal WWW-pages) much more interactive as they can contain links to more information related to the presentation and the user might choose to follow these at will. They may browse back and forth or explore the interactivity built into the slides.
- There is a lot of information available and it is very easy to `published` the slides from a presentation.
- It is very easy to create new presentations.
- There are WWW clients for virtually any platform.

The multicast Web - mWeb is a tool that allows users to distribute World Wide Web objects and pages in a scalable way. It also allows users to synchronize their browser to display the same page.
mWeb can be used to share objects addressed by URLs entered manually, or be configured to follow the selections made in a running browser. The later means that after the configuration, the user can select links just as it normally would be done and all listening members browsers will "follow" the sending members browser without any other interaction.

mWeb displays a list over viewed WWW objects by their corresponding URLs and a "listener" can go back and re-view already displayed objects. mWeb supports distribution of any WWW object that is retrieved using the Hyper Text Transfer Protocol - HTTP [11] which is the primary protocol for retrieving data on the WWW.

The mWeb application is developed at CDT and is further discussed in [12].

3.6 Recording and Playback of Sessions

A very big advantage of using digital network based tools for teamwork and communication is that the data can easily be recorded digitally and later played back. This functionality is added to the mStar environment by the multicast Media-On-Demand application - mMOD [13] which is also developed at the CDT. It allows recording of any public MBone session and later playback of that recording. Playback of recordings is requested using a Web interface. It records all media in a session, and they are all synchronized at replay.

mMOD supports recording of any type of IP-multicast based traffic, which later can be replayed. This allows for recording of any type of media or collaborative tool, even if the network protocol or tool used is not known. When the session is later replayed, the receiving tools will not see any difference between the replayed traffic and the original traffic.

During playback, the viewer has "random access" within the recording using a Java applet (see Figure 7), which also displays information about the length of the recording and the current playback point. The user can also index each recording automatically or by hand. Each index will act as a bookmark into the recording when later played back and the viewer can then, using these indexes, easily jump to interesting points in the recording.

![Figure 7, The mMOD VCR control applet.](image)

As playback of a recorded session is not different from a 'live' session, listeners can interact during the playback. This allows for interaction and commenting during playback. Recording and playback can e.g. also be used by users that can not attend a meeting, but are supposed to talk during the meeting, i.e. their talk can be recorded earlier and played back.
during the meeting and the listeners will not see any difference (other than that they can not ask any questions to the presenter at that time).

3.7 The multicast Tunnel - mTunnel

Another member of the mStar environment is a tool called the multicast Tunnel - mTunnel, which also have been developed at the CDT. mTunnel allows application-level tunneling of multicast traffic based on explicit user decisions. This means that the user has to explicitly choose which sessions to transport through the tunnel. This tool allows users to tunnel multicast traffic over network segments that normally do not support multicast (such as ISDN and modem connections).

mTunnel also allows the user to select if the data transmitted over the link should be “scaled” in different ways: e.g. audio can be transformed into an encoding that requires lower bandwidth (the GSM encoding requires only about 25% of the bandwidth needed by the better quality PCM encoding which is normally used); Video can be voice-switched, meaning that only the video from the current speaker is transmitted over the link; Audio from simultaneous active sources can be mixed together to create a single audio-stream; Video bandwidth can also be reduced by dropping a certain amount of data from the data-flow based frame information and not just packet information (this is possible because the protocol used to transmit the data is designed to withstand packet-loss and contains information about where new frames start).

The mTunnel application is further discussed in [14].

4. Developing New Teamwork Tools

To ease development of new teamwork tools, a service and application creation platform was developed. This platform consists of a generic agent architecture and an application-level network layer, the Tunable Multicast Platform - /TMP.

4.1 The Generic Agent Architecture

Each application is divided into agents, which is a program-module responsible for a single and independent task. Such a task can either be self-contained within the agent, or be implemented by integrating an existing tool. In the latter case, the agent acts as a mediator between:

- The external tool language, which is the set of commands that is specific to the existing tool. An example of such a language is the remote-control api used to control the Netscape World Wide Web browser.
- The internal tool language, which is the set of commands that every agent must understand and are distributed to instances of the tool to achieve the desired functionality. This is the Control Bus protocol (see below).

Examples of self-contained agents are mChat, mVote and mWB (discussed earlier), and an example of an integration agent (mediator) is the World Wide Web agent (part of mWeb) that controls external browsers.
4.2 The Tunable Multicast Platform - /TMP

The Tunable Multicast Platform - /TMP, is a package for multi-point communication in a scalable and reliable way over intra- and internets. The base of the library is an implementation of the Real-time Transport Protocol - RTP [15] for basic real-time data-transmission and loosely coupled membership-control. RTP is the primary protocol for distribution of multi-point real-time data on the Internet and is used in most Mbone application available today.

The library also contains implementations of two new protocols developed at CDT, Scalable Reliable Real-time Transport Protocol - SRRTP [16], and Scalable Reliable File Distribution Protocol - SRFDP. These protocols add support for reliable multicast and file distribution. SRRTP is based on the ideas from the SRM framework [17], where the main advantage is that every application that is a member of a session, not only the original sender, participates in the repair of lost packets.

/TMP also includes support for communication between agents using the Control Bus - CB [18]. CB is a simple, but still powerful, mechanism for messaging between agents. The message format allows addressing of a single agent or a selected group of agents. It is also used for inter-agent communication within applications that consist of more than one agent.

5. mStar Usage Scenarios

The mStar environment can be used in a number of ways to create a distributed teamwork environment, which scales very well to a large number of users and members.

The mStar environment is used daily at CDT to create a distributed electronic workspace, with users working from their homes, users located at CDT and users located at CDT involved companies. The mStar environment is currently used in a number of different ways:

Electronic meetings: where the environment is used as a replacement for the physical meeting room. Here, audio/video is normally used in conjunction with mDesk. These meetings are normally also recorded using the mMOD system which allows users to later retrieve the data and review what has been said during the meeting. It also allows people that do not have time to attend a meeting to join in and just listen with one ear when they have time. In the "physical environment" this is normally not done by busy persons as it takes time to go to the meeting rooms and they normally do not want to disturb by coming late or leaving early (they usually want to be near their telephone as well).

Electronic education: where some or all teachers/students are geographically separated. Tools used here are normally audio/video, mWeb for slides and mDesk for group assignments, questions and feedback. A number of different scenarios are used here:

- The teacher is talking to a group of students, who are all in the same room, but the presentation is also distributed electronically to allow interested parties to join in. This allows for better information dissemination to other students, non-students and employees that normally would not have the time to follow a complete course or would not fit in the room.
Some of the students are located far away and can not attend the physical lecture.

Several teachers are used within one lesson as complement to each other. This could of course be accomplished in the "physical environment" as well, but the problem of organizing several different teachers to be in the same room at the same time is overwhelming. Also, if one of the presenters can not be present during the lecture, the appearance can be recorded in advance and played back during the presentations, allowing for virtual presence.

Of course, all presentations are recorded digitally, which allows students to review lectures at a later time. See Section 6 for a further discussion on how the technology have been deployed and used for electronic distance education.

**Electronic corridor:** where users just talk with each other. Media used here is usually only audio/video. The audio is normally used as an open channel for questions and announcements, either directly to some member or to the whole group. By using it as an open channel, it creates group awareness through other users overhearing useful and interesting information. It also allows users to peek into other users' rooms to see if they are available and present before trying to talk to them.

## 6. The Education Direct Project

The Education Direct project is a major effort on exploiting new technology for electronic communication, and involves, in various ways, several hundred people. It was initiated during the spring 1996, and spent significant effort on getting a firm rooting among end-users outside the already knowledgeable kernel of technologists. The operational phase started during the fall 1996. The goals are as follows:

- Accomplish a significant increase in broad use and understanding of multimedia technologies on the Internet in the county of Norrbotten.
- Establish better direct contacts between CDT / Luleå University of Technology, and high schools and secondary schools, and SME's with respect to this technology.
- Illustrate the use of the technology by distributing a course on this actual subject, (Internet and Distributed Multimedia), to those people.
- Establish and test the infrastructure needed to accomplish this.

Among several longer-term goals, this is the first concrete step on implementing the vision of the electronic presentation and education environment, according to which all courses at the Luleå University of Technology and CDT will be possible to attend independently of physical location.

This is a need that is especially urgent in the county of Norrbotten, with its sparse population and large geographic distances, (approximately 400x400 km).

The project involves four kinds of people:

- **Technicians**, which must be able to manage computers and communication equipment to ensure continuous operation.
• **Advanced Users**, which should be able to utilize the technology for doing own productions, presentations, and information searches.

• **Simple Users**, which need to be able to use the technology to benefit from the information provided by others.

• **Technologists**, which in addition should have an overview of how the technology works, ongoing trends, and principles of the area.

The participants come from all over the county, many of which are high-school teachers, (as those will act as local technology transfer persons). Part of the material is also included in a student course at the Luleå University of Technology.

### 6.1 Results and User Feedback

The response to the initial part of the project was very positive by all participants (both inside and outside the CDT) and have led to a large demand for further courses and further use of the technology. The University wills together with the CDT distribute several courses during school year of 1997/98 and will continue the successful distribution of graduate courses, which also started during 1997.

A direct result of the successful dissemination of the technology to the lower grade schools, is a new project for teaching 12-13 year old pupils Spanish using electronic distance education. Participation in the course will be voluntary and the teaching will be done by non-university teachers that did not know anything about these tools or the technology in the beginning of 1997. This course would not be possible to be given without the support of the electronic distance tools, as the number of students at each school would be to few.

### 7. Privacy and Social Group Aspects

As every room is equipped with a camera, and the contents of the camera is distributed to viewers over the network, readers might raise a concern (most newcomers do) about the privacy off the person in the room, but that person always have the choice to turn of the camera. Persons that have just begun to use the system often switch off their camera, as they feel disturbed by the transmission, but they soon get used to it, just ignore it, and leave it on all the time.

Every session does not need to be public, of course. A group of users can create a private session that will not be announced to the rest of the world but will only be seen by invited users.

Using this environment, a distributed work area can be created which allows users to work together although they are not physically located at the same location. It also creates an environment where persons working over a distance can get a group connection feeling and do not feel isolated (which often happens otherwise), as they can see other persons working. The group feeling also arises even if the persons do not work together or even know each other.
8. Implementation

All tools developed at the CDT within the mStar environment are Java based (with the exception a GSM encoding library in mTunnel which is implemented in C for efficiency reasons and an implementation that this library was already freely available on the Internet).

The advantages of using Java as a development language are many, including the development of these tools would had taken much longer time if done in for instance C or C++. Another large advantage is that it makes the tools truly platform independent.

The mStar environment runs on most UNIX platforms and under Windows95/NT4.0. It does not run under MacOS due to the lack of a Java1.1 compliant virtual machine needed for running the programs written for this version of the Java language.

9. Further Issues

There are a number of open issues and missing applications. One example is how to locate users and how a user can leave a message about her whereabouts (compare to a note on the door to their office or a voice message in the telephone-system). Another tool of high interest is a digital answering machine.

The usage of digital recordings has opened a number of issues related to annotation and editing. There are also a number of issues around mMOD regarding load balancing between servers and how media available on different servers can be integrated into a common view to the user. The transformation components of mTunnel should of course also be integrated into mMOD allowing users on low-bandwidth links to receive recordings.

The questions of privacy and social group aspects are very large and currently mainly uninvestigated with this kind of tools.

Another open issue is better floor control, meaning that is allowed to transmit data into an active session and how this is supposed to be controlled.

Currently the video quality is low (compared to broadcast TV but high if compared to many other video-tools currently available on the Internet), and this is of course also an interesting issue to approach.

10. Summary and Conclusions

This paper presents the CDT mStar environment, which creates an environment for true Scalable Distributed Teamwork.

A number of real-time synchronous media were presented, as part of this environment: audio/video, whiteboard, chat, vote and shared presentations. These are all desktop based, as the user should not have to move to another location just to have a short talk with a colleague.
Audio/video was realized using the already existing Vic and Vat, but the other tools had to be developed from scratch, which was done at CDT. The whiteboard, chat and vote functionality was added to the mStar environment by the mDesk application. mDesk includes a distributed shared digital whiteboard called mWB, which can be used to review text-documents and images, joint drawing, copying/moving of drawn objects and loading/saving of documents/drawings. mDesk also includes mChat, which is a text-based chat tool where members can chat with each other, and mVote, which is a tool for distributed voting. Distributed presentations using the World Wide Web is added by the mWeb application.

Two other members of the mStar environment were also presented: mMOD and mTunnel. mMOD is a VCR-like tool which allows recording and playback of the electronic work done within the group environment. mTunnel is an application for handling bandwidth on narrow links (such as ISDN and modem) in the network. It allows for scaling and transforming of the network based audio and video in various ways.

Central design issues related to all these tools are that they should be desktop oriented (there should be no need to go to a specially designed and equipped room), they should scale well to a large number of users, they should all be network based, and they should be symmetric, meaning that no tool is central and that every user can do the same thing with their copy of the tool as anyone else in the session.

The mStar environment is currently being, and has for some time now been used in a number of different ways to support distributed teamwork. Three major usage scenarios where identified: The electronic corridor, where the system is used to exchange information between users as if they would when they meet in the corridor; electronic meetings, where the system is used as a replacement for the physical meeting room with the advantage that everything can be recorded and later reviewed; and distributed education, where teachers/students do not have to be located at the same physical location.

With the mStar environment, a Scalable Distributed Teamwork environment that is really used daily (24 hours a day), have been created. It allows users to work from their homes or other offices connected to the Internet without risking to lose contact with their main work group and project team.

References


Acknowledgments

There have been a large number of persons involved in developing and using this system, but we would especially want to thank, Håkan Lennestål, Johnny Widén and Ulrika Wiss, CDT, for comments, nice encouragement and interesting feedback.

This work was partially done within Esprit project 20598 MATES, which is supported by the Information technology part of the 4:th Framework Program of the European Union. Support was also provided by the Centre for Distance-spanning Technology (CDT). The Education Direct project is supported by ”Stiftelsen för Kunskaps- och Kompetensutveckling” - the Foundation for Knowledge- and Competence-development.
Paper E

Lightweight Application Level Multicast Tunneling using mTunnel
Lightweight Application Level
Multicast Tunneling
using mTunnel

Peter Parnes, Kåre Synnes, Dick Schefström
Luleå University of Technology / Centre for Distance-spanning Technology
Department of Computer Science
971 87 Luleå, Sweden.

ABSTRACT

This paper presents a system, called mTunnel, for application level tunneling of IP-multicast traffic in a lightweight manner, where the end-user is responsible for deciding which MBone-sessions and which IP-multicast groups to tunnel. mTunnel is primarily designed for easy deployment and easy-to-manage tunneling. Information about currently tunneled sessions and control of mTunnel is provided through a Web-interface. To save bandwidth, tunneled streams can be transcoded on the data-level; and traffic sent through the tunnel can be compressed by grouping several packets together and using statistical compression. The overall bandwidth reduction achieved can be between 5 and 14% depending on the traffic type.

Keywords: Multicast tunneling, packet and header compression, rate control, automatic media transcoding, MBone, Java.

1. Introduction

On the Internet today, more and more applications are starting to use IP-multicast [1] (as of now multicast will refer to IP-multicast) for distribution of primarily real-time data such as audio and video. A growing number of more conventional applications are starting to utilize the power of multicast. Multicast is a special form of addressing and sending data on the Internet where the sender sends the data to a so-called multicast group instead of sending the data directly to a particular host. Hosts that want to receive data from the sender joins the multicast group and the network routing protocols see to that the traffic gets to the new receiver. When the receiver no longer wants to receive the data his host will notify the network of this and the data will no longer be forwarded to that part of the Internet (with the exception that there might be other receivers on the same local network that want to continue to receive the data). Note that the sender does not need keep track of who wants to receive the data but this is done by the network itself.

Multicast has a number of advantages, such as its inherent scalability of data propagation where data is only copied in the network where needed. It also has built in robustness as it does not depend on any central resources such as reflectors or mirrors for data propagation.
However, a major problem with multicast is that it is not part of the core IP version 4 family and by that only a small part of the Internet actually supports it.

To ease the development of multicast related protocols, a virtual network called the Multicast Backbone, MBone [2], was developed. It consists of tunnels between nodes that act as virtual routers, exchanging multicast packets encapsulated in unicast packets. Encapsulation means that multicast packets are received at a node that is connected to a multicast enabled network, repackaged into a unicast packet, sent over the portion of the Internet that does not support multicast to another node that is part of a multicast network.

This second node removes the unicast information and resend the packet on its multicast network. The MBone has existed for several years now and is slowly being deployed in production networks. A remaining problem is that unfortunately most dial-up links do not support multicast. This means that there is still a large need for tunneling of multicast over such links. Characteristics of these links are that they usually only connect a single host or a single local network with a few hosts and that these dial-up links most often only have a limited amount of bandwidth available.

Another reason for using tunneling of multicast is firewalls. At some companies, experiments have shown that it can be very hard to convince the system administrators to open the company's firewall for multicast traffic, but interestingly easy to convince them to open a few ports for TCP and UDP traffic to a specific host behind the firewall. There are a number of problems and questions related to tunneling of multicast which are:

- How should the tunneled packets be encapsulated when sent over the tunnel to waste as little bandwidth as possible?
- How much control information is needed to be exchanged to maintain the tunnel operational?
- Can the tunneled data be transformed on the IP-level (without any knowledge about the content) before being sent through the tunnel to lower the bandwidth needed for the tunnel?
- Can the tunneled data be transformed on media-level (with knowledge about the content) before being sent through the tunnel to lower the bandwidth needed for the tunnel?
- Can the deployment of tunnels be made simpler and more lightweight than available previously?

To target these problems and questions an application called the multicast Tunnel, mTunnel was designed and developed. These problems and questions and this application are the focus of this paper.

1.1 Related work

The primary and most popular application for tunneling multicast traffic on the MBone is the MRouted [3] application. This application acts as a virtual multicast router and includes a lot of routing functionality. This means that the application has to have access to privileged information within the operating system it runs on and it also needs special routing support in the kernel.
This means that the application can only be used on machines that contain the required routing functionality (such as hosts running variants of the UNIX operating system\(^1\)). Since, MRouted is a virtual router it implements and utilizes multicast routing protocols to control its tunnels. These protocols often exchange a lot of control and routing information which can be a problem on low-bandwidth links (Note that the mTunnel application presented in this paper is by no mean a replacement for the core MRouted tunnels in the MBone. Instead it should be seen as a complement to allow more users to use multicast applications.).

liveGate [4] is another application for tunneling of multicast traffic, which was developed concurrently and independently of mTunnel. liveGate supports basic lightweight deployment of multicast tunnels and tunneling of multicast traffic, but does not currently support any traffic modification to lower the bandwidth.

A number of different tools and gateways for modification of real-time multicast traffic also exist, such as the Robust Audio Tool - RAT [5] which can convert audio data between different formats and the RTPGW [6] which focuses on that bandwidth is not uniformly available on the Internet for audio and video transmissions. It can convert video between different formats and it also supports some basic audio transcoding.

The rest of the paper is structured as follows: Section 2 gives an overview of the mTunnel application; Section 3 discusses the use of non-lossy compression schemes for lowering the bandwidth; Section 4 describes rate control mechanisms for data flows; Section 5 examines extra delay introduced by the mTunnel application; Section 6 presents the implementation and with Section 7 we conclude the paper and give a summary.

2. The mTunnel application
Tunnels always have two endpoints between which traffic is tunneled. For that reason there must always be two instances of mTunnel running, one at each end of the tunnel. The basic functionality of mTunnel is very simple: listen to a number of multicast sessions, encapsulate all traffic received on these sessions, send the traffic through the tunnel, on the other end decapsulate the traffic and resend it locally using multicast (see Figure 1).

![Figure 1, Overview of the data path of an incoming packet through mTunnel. The three boxes in the center represent optional traffic modifiers.](image)

\(^1\) Microsoft Windows NT5 is supposed to support routing and contain the needed functionality to run MRouted.
mTunnel runs as a user process at application level to allow easy and lightweight deployment. This means that the user does not need any special privileges to run the program. All the mTunnel application requires from the operating system is that it supports multicast. The main user interface to mTunnel is through a built-in Web-server which allows users to get feedback about currently tunneled sessions and easily start tunneling of either publicly announced MBone sessions or private sessions where the user specifies the needed session information manually.

The tunneled data is sent over a UDP unicast “connection” between the two endpoints and all data is sent over the same port number. This allows for easy tunneling through firewalls (as the systems administrator only has to open one port in the firewall). Control information is exchanged on a TCP connection between the two tunnel ends for reliable messaging.

Each multicast packet is encapsulated by adding a special trailer. The reason for using a trailer instead of a header (which is the case in most protocols) is twofold: it minimizes the number of required copy operations per tunneled packet (a header would impose that the incoming data would need to be copied to make room for a header) and that the IP/UDP/RTP² headers remain intact, which in turn allows lower level header-compression schemes to be utilized [7].

To lower the amount of encapsulation data sent over the tunnel, flow identifiers are used instead of sending the multicast-group and port-number with each tunneled packet. By using a flow identifier of size 1 byte, the trailer is reduced by 5 bytes which lowers the bandwidth with e.g. 1 Kbps for a constant 40 ms PCM audio flow (71 Kbps).

2.1 Data translation

The required bandwidth for a tunneled session may be lowered by transcoding the data. mTunnel includes a translator, that may translate the traffic in various ways. Currently the translator includes a recoder, that translates between different audio encodings; a mixer, that mixes several active streams into one single stream; a switch, which only forwards packets from a certain source based on another stream (for instance, video packets are only tunneled for the group member that currently is speaking - voice switching). The last part is the scaler, that rescales the traffic by dropping packets based either on just packet count or by taking frame information into account. The latter makes a big difference on the result if a video encoding such as H.261 is used where one frame often spans more than one RTP packet. The different modes of the translator can also be pipelined, e.g. an audio session can be both mixed and recoded. All these different translation methods are lossy meaning that they actually remove information from the data stream. This transcoding allows users behind low bandwidth links to join sessions with a total higher bandwidth than the one available on the low bandwidth link.

The MRouted application (as presented in Section 1) connects multicast capable islands of networks to form the MBone. These network become an integrated part of the MBone.

² RTP, Real-Time Transfer Protocol [8] is the main protocol used on the MBone for transmitting real-time data.
virtual network and all traffic that is requested is tunneled to and from them. The decision which sessions to tunnel, is based on requests made by the multicast aware applications that the users on the other side of the tunnel starts. The messaging between hosts and multicast routers is done using a protocol called the Internet Group Management Protocol - IGMP[9].

As soon as a multicast aware application starts and requests data for a specific multicast group an IGMP message is sent to the nearest multicast router which in turn will see to that the corresponding multicast traffic is propagated to that particular network.

This model works very good if the bandwidth is not limited, as several sessions can be tunneled at the same time. However, if the bandwidth is limited (like over analog modems or ISDN-links) the users have to quit their MBone applications to stop a session from being tunneled (i.e. if an application “wants” multicast traffic for a special multicast group, usually the only way is to quit the application to stop the traffic from being tunneled). Also, if several users share the same narrow link, it might be complicated or even impossible to coordinate which sessions to tunnel (several users join different sessions at the same time).

tunnel instead uses a *user decided session selection model* where the local users explicitly have to choose which sessions to tunnel. This has several advantages, such as making the end users aware of other currently tunneled sessions and removing the need for users to quit their multicast tools to stop the tunneling of specific multicast groups.

### 2.3 Transmission loops and Time To Live

tunnel is designed to connect a network or a single host that is currently isolated from the MBone, to the MBone. However, if both ends of a tunnel are directly connected to the MBone, a transmission loop can occur if the tunneled MBone-sessions are not chosen carefully. tunnel therefore does not forward packets through the tunnel if the sender matches the other end of the tunnel. Unfortunately, if two separate tunnels are deployed that together create a loop, packets will be forwarded over and over again.

If tunnel suspects that a loop has occurred (the packet rate through the tunnel suddenly raises dramatically), it stops tunneling traffic, sends out a special probe-packet and waits for the probe-packet to be received again. If the probe-packet *is* received, all current tunneling remains stopped and users of the system are notified through the Web-interface. If the probe-packet *is not* received, the process is repeated a couple of times, because the probe-packet could have been lost on the way due the to best-effort nature of UDP-packets.

When packets are sent on the MBone today, their reach is limited by a so-called *Time To Live (TTL)* value. For instance, if a user wants to send multicast packets to the local network only, the packets are sent with a TTL of 1. In the current standard version of the socket interface under Unix and Windows, there is no way for a user application to get information about the TTL of an incoming packet. Due to the socket interface, and the fact that tunnel runs as a user-application, tunnel can only forward packets based on the

[^3]: The way an application speaks with the operating system and the network.
TTL value specified when the session was created. Unfortunately, this means that if a user sends traffic in an announced MBone-session with a lower TTL than the announced TTL, the local packets will be “amplified” and retransmitted with a higher TTL than intended by the original sender. Currently the only available answer to this problem is to make the users of mTunnel aware of the problem.

3. Non-lossy compression of tunneled data

The amount of data sent through the tunnel can be lowered by compressing it using a non-lossy compression scheme. Normally, this is not done on single real-time flows as the encoding used to encapsulate the data usually includes some kind of compression scheme as well. But, in the case where several different data flows are concentrated at a single point in the network (such as the tunnel ends), the redundancy between the different flows can be used for compressing the data. This section discusses how data compression have been utilized in the mTunnel application and how much the bandwidth is reduced for different kinds of data flows.

The more uncompressed data available, the more it can be compressed due to the higher probability of similar and redundant parts. This is generally not a problem when compressing data files as all the required data is already there, however, when compressing real-time data flows the goal is to keep the delay low (i.e. not to buffer the packets too long). Another goal is to keep each resulting compressed packet independent, so if it gets dropped in the tunnel, following packets may still be decompressed and do not depend on previous packets. To be able to compress the data flows, packets have to be buffered before being sent through the tunnel. Secondly, there must exist an efficient way of recovering from unsuccessful compression attempts, as a group of packets might actually generate more compressed data than they contained from the beginning.

To allow for compression, incoming packets are grouped together. This grouping can be seen as a first step of compression as each tunneled packet has a cost in number of bytes sent over the tunnel due to the IP/UDP header each packet include (i.e. by grouping several packets together only one set of headers is needed per sent grouped packet). The cost of the per packet header is a minimum of 28 bytes and by grouping n (2 or more) packets together, (n -1)*28 bytes may be saved. This might not sound as much, but remember that most data flows consist of very small packets, e.g. a 40 ms PCM audio flow consists of packets containing 320 bytes of data plus headers. This first step of compression is referred to as header suppression. If the available grouped data cannot be compressed, then it is sent grouped as one large packet. The algorithm used is:

```c
for each packet do
  if not packer.willFit(packet)
    packer.generateAndSend()
    packer.add(packet)

generateAndSend:
  if compressed size < original size
    sendGroupedAndCompressed()
  else
    sendGroupedAndUncompressed()
```

100
Five different sessions (including several different data flows) have been examined with mTunnel to see how good they can be compressed. The measurements were done by saving information about each compressed packet within the mTunnel application and the measurements were done over a time period of 1 hour. In none of the tests lossy transcoding was performed and to be able to repeat the exact traffic, sessions were played back using the mMOD system [10]. Table 1 presents how much bandwidth can be saved by using tunnel compression on the different data flows. The table shows that for normal data flows a compression ratio of about 5-9% is achieved. The reason for the fourth session getting a higher compression ratio (14%) than the rest is that it includes a whiteboard stream which consists of many very small packets which in turn contain a lot of redundant information. In session 5, hardly any compression is achieved due to the fact that the data flow consists almost entirely of packets with a constant size (1024 bytes + headers) containing already heavily compressed MPEG coded audio data [11].

<table>
<thead>
<tr>
<th>Session</th>
<th>Original Bandwidth Kbps</th>
<th>Total Saved % Kbps</th>
<th>Saved due to compression</th>
<th>Saved due to header compression</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Local Lecture</strong></td>
<td>198</td>
<td>6.11%</td>
<td>3.05%</td>
<td>3.06%</td>
</tr>
<tr>
<td>audio, video, HTML using mWeb[12]</td>
<td></td>
<td>12.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Electronic Corridor</strong></td>
<td>170</td>
<td>9.23%</td>
<td>4.60%</td>
<td>4.62%</td>
</tr>
<tr>
<td>Audio, video</td>
<td></td>
<td>17.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Meeting</strong></td>
<td>229</td>
<td>5.15%</td>
<td>2.08%</td>
<td>3.08%</td>
</tr>
<tr>
<td>audio, video, mDesk [13]</td>
<td></td>
<td>12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Global Lecture</strong></td>
<td>192</td>
<td>14.3%</td>
<td>10.9%</td>
<td>3.32%</td>
</tr>
<tr>
<td>audio, video, WB</td>
<td></td>
<td>31.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Constant MPEG</strong></td>
<td>131</td>
<td>0.128%</td>
<td>0.0363%</td>
<td>0.0920%</td>
</tr>
<tr>
<td>using mIR [11]</td>
<td></td>
<td>0.164</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 1, Overview over compression results for different sessions using non-lossy compression.*
4. Priority and rate control of data flows

As mTunnel is designed to run over links with narrow bandwidth it is important to support rate control for the total tunnel and for each tunneled session.

This can be used for e.g. prioritizing audio over video by reducing the amount of bandwidth available for video. By reducing the total available bandwidth for the tunnel, other types of traffic might be given more room on the bandwidth restricted link. A higher priority means that if congestion occurs, the traffic belonging to the session with higher priority will be forwarded before traffic for a session with a lower priority.

Each tunneled session can have its own forwarding priority and by default, all sessions have the same priority. Using the Web-interface, the priorities can be changed individually. Priorities for one or several sessions can also temporarily be locked, meaning that no other session can be given a higher priority than that session as long as it is locked (note that, locking does not mean that the session gets a higher priority, only that its priority can be exceeded by another future session). This is useful if an important electronic meeting is conducted over the tunnel and the participants do not want to be disturbed by another user who wants to watch some other MBone-session.

Priorities, can also be configured in mTunnel, based on a number of different variables in the session: the media-type, the multicast address and port, the used bandwidth, the name and the description. This allows for advanced selection of priority schemes, that enables a user to participate in sessions even if the total needed bandwidth is not available.

To control the data flows, a so called bound token bucket [14] scheme is used. The main idea of this scheme is to have a bucket of tokens that is filled up regularly depending on the available bandwidth for that flow. Each token corresponds to one byte of available bandwidth and when a packet is forwarded the number of tokens are reduced by the corresponding size of that packet. The bucket has a limited size, meaning that there can never be more than a certain amount of tokens available. An infinite number of tokens would make it possible for a bursty data flow to overrun the available bandwidth. This might still happen in the bounded bucket case, but the impact is much less severe if the max size is relatively low. (A hard limit against overruns can be achieved by adding a leaky bucket [14] as well).

5. Data delay

The use of tunneling with its different schemes for reducing the bandwidth and the bursts in bursty flows has the side effect of introducing extra jitter and delay in the tunnelled traffic. This section examines how much extra delay is actually introduced by the tunnel. The introduced delay was measured on the global lecture (row 4 in Table 1). The measurements were done by measuring the round-trip-time (RTT) of a data flow between two hosts on different networks, one sender and one echo-host which bounces the received traffic back to the sender. Tests were conducted over two types of networks, Ethernet and ISDN.

For each type of network three tests were conducted: one where the two hosts had direct connectivity through a router (using only unicast in the ISDN case), a second one where the
traffic was tunneled between the two networks using mTunnel with no compression or transcoding and a third one using mTunnel with compression turned on. For each of the tests the round-trip-time (RTT) was measured. Table 2 shows the mean delay (RTT=2) for the six cases. As seen in the table, the delay gets quite high when using mTunnel with ISDN and compression turned on. This high delay could make synchronous conversations uncomfortable as the RTT would get up to around 276 ms which is above the usually recommended 200 ms maximum for conversations.

<table>
<thead>
<tr>
<th>Network type</th>
<th>Direct contact</th>
<th>Tunneling without compression</th>
<th>Tunneling with compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet</td>
<td>2.00</td>
<td>14.4</td>
<td>74.5</td>
</tr>
<tr>
<td>ISDN</td>
<td>52.0</td>
<td>74.1</td>
<td>138</td>
</tr>
</tbody>
</table>

Table 2, Mean delay (RTT=2) of traffic for the different measured cases (ms).

6. Implementation and Status

The current prototype is implemented in the platform independent Java language (version 1.1), except some parts of the audio transcoder that are implemented in C for efficiency reasons. The non-lossy data compression is done using the built in version of the ZLIB [15] compression library in the Sun Java environment. The audio recode functionality is currently only available on Sun/Solaris, but the rest has been tested to work under both Unix and Windows95/NT4. All tests and measurements mentioned in this paper were done under Sun/Solaris.

mTunnel is currently being used in three different ways: to connect different parts of a large software company’s intranet, to connect computers at users homes, and to connect industry networks to the MBone. More information about the current version, status of mTunnel, earlier publications and the program itself can be found at [16].

6.1 Further issues

There are a number of open security issues not yet addressed such as encryption of the tunneled data and IP-spoofing. [17] proposes a solution for the IP-spoofing problem which should be further examined. The user interface currently does not include any user authentication, meaning that any user can modify the tunnel.

The translator should also be extended to support a larger variety of encodings. Another issue currently not addressed, is the possibility of sharing media translators between two or more concurrently running instances of mTunnel to save CPU usage by only translating the same stream at one host. Other types of non-lossy compression schemes should be investigated as well. A limitation in the current implementation is that a TCP connection is used for exchanging control information and a UDP “connection” for the actual tunneled data. If these two could be combined deployment together with firewalls would get easier as only an UDP port would have to be opened for mTunnel and not both a TCP and a UDP port as it is the case today.
6.2 Evaluation
In the introduction a number of questions and problems were introduced. This section presents how mTunnel target these.

- How should the tunneled packets be encapsulated when sent through the tunnel to waste as little bandwidth as possible? The encapsulation of the packets in mTunnel is done by only including minimal extra information such as multicast group and port. To lower the overhead flow-identifiers and packet grouping is used.

- How much control information is needed to be exchanged to maintain the tunnel operational? mTunnel tries to minimize the amount of control information sent through the tunnel and no periodic update messages are sent.

- Can the tunneled data be transformed on the IP level (without any knowledge about the content) before being sent through the tunnel to lower the bandwidth needed for the tunnel? mTunnel can compress traffic using statistical compression without knowledge about the type of traffic currently being tunneled.

- Can the tunneled data be transformed on media level (with knowledge about the content) before being sent through the tunnel to lower the bandwidth needed for the tunnel? mTunnel allows traffic to be translated in a number of different ways such as: media can be recoded to another encoding; several active streams can be mixed; stream can switched based on activity in another stream and the streams can be rescaled by dropping parts of the traffic based on the type media in the stream.

- Can the deployment of tunnels be made simpler and more lightweight than with earlier available software? mTunnel only require that the host operating system supports multicast and can therefore be run on almost any operating system. The system includes a Web-interface which allows users to easily configure mTunnel and select which sessions to tunnel. mTunnel is implemented in Java which means that it runs on any platform that supports Java.

7. Summary and Conclusions
This paper presents a system for allowing users to easily connect to the MBone infrastructure and to connect different isolated multicast capable networks. mTunnel gives users easy access to information about currently tunneled sessions through a Web-interface, which also allows for easy configuration of existing and future tunneled sessions. mTunnel does not start tunneling of MBone-sessions based on current multicast group activity and IGMP messages, but instead makes the user responsible for deciding which MBone-sessions are to be tunneled. This allows for a user decided session selection model where tunneling decisions are left explicitly to the user.

To save bandwidth, data streams can be transcoded in four different ways: audio can be recoded to an encoding that requires lower bandwidth, several simultaneous audio streams can be mixed into a single stream, streams can be switched based on another stream, and streams can be scaled by dropping certain parts of the traffic. The traffic can also be
compressed on the IP-level which saves about 5-14% bandwidth. Unfortunately, measurements show that the compression adds a high extra delay to the tunneled traffic.

mTunnel does not require any special features in the operating system other than general multicast support and it runs as a normal user application. This means that it is very easy to deploy. mTunnel also tries to minimize the amount of control traffic exchanged through the tunnel and to minimize the cost of encapsulation of data by using for instance flow identifiers instead of adding group and port information to each tunneled packet. Note that the aim of mTunnel is by no mean to replace existing software for core tunneling in the MBone, such as MRouted, but should instead be seen as a tool for connecting more users to the global multicast network.

mTunnel is currently being used in three different ways: to connect different parts of a large software company's intranet, to connect computers at users homes, and to connect industry networks to the MBone. More information about the current version, status of mTunnel, earlier publications and the program itself can be found at [18]. The usage of mTunnel has shown and proven that it is useful and that there is a need for this kind of applications.

References

<URL: http://www.cdt.luth.se/~peppar/progs/mTunnel/>.
draft-finlayson-utmp-01.
<URL: http://www.cdt.luth.se/~peppar/progs/mTunnel/>.

Acknowledgements

Thanks to Mattias Mattsson, Fredrik Johansson, Serge Lchapelle, Håkan Lennestål,
Johnny Widén and Ulrika Wiss, all at Centre for Distance-Spanning Technology, and R. P.
C. Rodgers, U.S. National Library of Medicine, for interesting comments, encouragement
and feedback.

This work was done partly within the Esprit project 20598 MATES, which in turn is
supported by the Information technology part of the 4th Framework Program of the
European Union. Support was also provided by the Centre for Distance-spanning
Technology (CDT).
Paper F

A Framework for Management and Control of Distributed Applications Using Agents and IP-multicast
A Framework for Management and Control of Distributed Applications using Agents and IP-multicast

Peter Parnes, Kåre Synnes, Dick Scheffström
Luleå University of Technology / Centre for Distance-spanning Technology
Department of Computer Science
971 87 Luleå, Sweden.

ABSTRACT

As more and more applications on the Internet become network aware the need and possibility to remotely control them becomes larger. This paper presents a framework for control and management of distributed applications and components. This is done using IP-multicast and an agent based application architecture. The target of the framework is to allow for resource discovery of both controllable elements and available control points in these elements as well as real-time control. All this is done in scalable and secure way based on IP-multicast and asymmetric cryptography. The presented framework is also independent of the underlying transport mechanism to allow for flexibility and easy deployment. The framework bandwidth usage and introduced control delay is presented. Details on the reference implementation of the framework and example usage scenarios where the framework is used to create bandwidth adaptive applications and better group awareness is also presented.

Keywords: IP-multicast, distributed management, control, secure messaging, reliable multicast, distributed applications, intelligent agents, quality of service management, Java.

1. Introduction

With the current increase in the number of deployed distributed applications the need for control and management grows. A central issue in any computing environment, both in academia and industry, is how to control and manage running applications. This is especially applicable and important to the more and more used application family of IP-multicast [1,2] based distributed real-time applications for primarily desk-top conferencing, distance education and media broadcasts with many simultaneous users.

These distributed applications usually utilize the available bandwidth more than traditional single user applications and they are usually more sensitive to large delay and jitter in the
network. It is therefore very important that real-time media applications do not compete with each other for the available band-width but instead co-operate and try to utilize the bandwidth in the best possible manner.

When users start using real-time distributed applications, the risk for users doing the “wrong thing” and flooding the network with too much data also increases. Historically, this has been handled by locating the application, host and responsible user that generates the extra network traffic and asking the user to either terminate the application or lower its network usage (or in a UNIX environment, the system administrator may even just log into the host and terminate the application in question). The issue of finding the responsible user becomes more complicated when distributed applications are used in confederation between several organizations and over the Internet.

We have found that there is a need for a control framework where applications can be remotely controlled. The use of band-width might be one of the most important issues as that usually affects many users. Other important control scenarios include remote user support where the support people can get information about how an application is configured and can remotely change this configuration.

This paper presents and discuss a general framework for management and control of distributed applications. The target is that the framework should not end up in one isolated implementation but instead several different inter-operable implementations should emerge over time.

The rest of this section presents an overview of the distributed mStar environment from where much of the work presented in this paper have come, current problems and related work. Section 2 presents the new control and management framework. Section 3 presents how the framework can be and is currently used. Section 4 presents the reference implementation of the framework including the mManager application and some future work. The paper concludes with a summary and discussion in Section 5.

1.1 The mStar Environment
Since 1995 we (CDT) have been developing the mStar environment [3] which is an environment for scalable and effective distribution of real-time media and data between a large number of users. mStar is a shared environment that can be used for a number of different distance-spanning activities such as net-based learning (distance-education) and distributed collaborative team-work. It includes support for a number of different media including audio, video, shared whiteboard, distributed presentations using the World Wide Web and much more. It also supports on-demand recording and playback of sessions using either unicast or IP-multicast. The idea and need for a management framework came from the daily usage of the mStar environment at CDT. (Note, that the mStar environment is now being commercialized and sold under the name Marratech Pro by the company Marratech AB in Sweden.)

1.2 Current Problems and Framework Requirements
When a distributed desktop conferencing application, such as the mStar environment, is deployed in a large organization; a number of new management and control issues evolve.
The administrators need to be able to get information about and control: which users are part of which conferencing sessions, which media in each session do they currently have active (i.e., which media are they currently receiving), if the user is currently transmitting any data within a session and then with which settings (e.g., for video transmission the settings in question would be the bandwidth currently used, frames per second or codec/video-format). If this information is available it will allow the administrator to control both session membership (e.g., kick out unauthorized members) and the total bandwidth used by this group of applications. This means that the network administrators can control the total amount of bandwidth used by each user and session explicitly.

To be able to monitor and control running applications, it is necessary to get feedback on what applications users are currently running, which version of the specific applications they are running, and the current configuration of these applications. Secondarily, it is necessary to be able to remotely control the applications. These problems can be divided into two major groups, information reports from users and remote control of applications.

The identified problems lead to a number of requirements:

- The framework have to support large groups of applications, both for reporting and control.
- The framework have to support efficient control of groups of applications without the need to send control messages to each application explicitly.
- The framework have to protect users from unauthorized control of their applications.
- The framework have to allow developers to easily insert control access points in their software.
- The framework should be as independent of the underlying software and transport technology as possible.
- The framework should allow for scalable resource discovery of both available control objects and controllable variables and methods in these objects.

This paper focuses on presenting a new framework for addressing these problems and requirements.

1.3 Background Information and Related Work
This section presents related background information and some selected related work.

1.3.1 IP-Multicast Applications
Traditionally, distribution of multimedia data on the Internet to a group of users have been done using unicast, either by each multimedia application sending one exact copy to every receiver or by pushing the problem into the network and using a so called reflector which duplicates streams to every registered receiver. This means that duplicate data will be sent if the path between the sender and its receivers share common links in the network.

The power of IP-multicast is that data is only copied in the network where needed when sending the same data to several different receiving hosts. IP-multicast traffic is sent using UDP which is best-effort and in turn means that packets can be lost in the network. This
might be a problem in control situations as the manager wants to be assured that sent control messages actually are delivered to the destination. This problem is further discussed in [4]. A drawback with IP-multicast is that it requires support from routers in the network to handle this special kind of traffic. If the routers in question are fairly new, it might be simply a question of turning on the support in the router software. To summarize, the IP-multicast solution saves large amount of bandwidth in the network.

1.3.2 Simple Network Control Protocol - SNMP
The Simple Network Control Protocol - SNMP [5, 6] is designed for control of network elements and basic applications. It is designed with a 'polling' architecture in mind meaning that the managing software have to request information from each element to be monitored. This architecture allows managers to get information from a monitored object and set variables in that object.

SNMP includes support for so called trap’s where a manager can request to be notified when some predefined situation occurs. A restriction with these trap’s are that they cannot be defined dynamically but the manager has to rely on predefined trap’s. Note, that although this trap mechanism exists the normally used part of the SNMP mechanism is still only the “get/set” functionality.

As SNMP is designed to control a single element at a time, it is not really suitable for controlling large groups of real-time applications. Another important aspect not supported by SNMP is resource discovery. Further, every user that want to control and fetch information from a device have to have a corresponding definition document called a management information base to know what variables are actually available in the device to be controlled.

1.3.3 The Service Management System - SMS
The designers of the Service Management System - SMS [7] argue that the poll-architecture of SNMP causes managers to often notice problems too late as the object in trouble cannot notify its manager about its condition. The creators of the SMS system try to approach this problem by creating an architecture where each managed object is encapsulated by a SMS wrapper which marshals commands to and from the managed object. The wrapper has a SMS agent to aid it with automatic handling of certain situations. This SMS agent can be controlled by a set of rules (defined using the so called Service Management Agent Programming language - SMA) to react on information provided by the SMS wrapper.

Although this architecture seems promising it does not include support for resource discovery, information reporting and scalable control of large groups of applications.

1.3.4 The Conference Control Channel Protocol - CCCP
Reference [8] presents the Conference Control Channel Protocol - CCCP which is a protocol for control of distributed multimedia applications. It consists of a text based message protocol for primarily control. The CCCP is similar to the work presented in this paper but differs in that we focus on a wider range of applications than just the group of
conferencing applications and not only real-time control but also scalable reporting of information.

Reference [9] presents a similar message platform but that one is even more constrained and only target the problem of how to communicate between similar applications within a single host.

1.3.5 Java Specific Platforms for Distributed Applications
There exits a number of proposals for Java based architectures for distributed applications such as the InfoBus [10], JavaSpaces [11] and iBUS [12]. Some are very promising and flexible but they all make one large assumption on their programmatic environment and that is that they are very tightly integrated with the Java programming language and its runtime environment. They all assume that they have access to features that are very specific to the Java environment, such as Java-events and/or serialization (binary representation of Java runtime objects). This assumption makes these frameworks virtually unusable in other environments.

1.3.6 Other Potential Control and Management Technologies
Several other technologies could be used as the underlying mechanism (such as Corba [13], MPEG-4 DMIF [14], and SS7 [15]) but during our investigations we have found that all of these are either not scalable enough (centralized solutions where the central point becomes a bottle-neck) or to specific to their original design domain.

2. The Control and Management Framework
The purpose of proposing a new framework is to support developers in the process of creating a new kind of applications that are network aware and fit better into the global Internet. This framework includes a set of building blocks that allows for scalability in resource discovery, information reports and real-time control.

As presented earlier there are a number of problems involved in how to control and manage real-time distributed applications. This section presents a new control and management framework for addressing these problems and requirements.

2.1 Information Reports
To be able to request information from currently running applications there must exist a platform that allows for reports to be sent in a scalable way. With scalable, we mean that the solution found should be usable within sessions with a few users and within sessions with a large number of users, as well as sessions that run locally and over wide area networks. The amount of bandwidth needed in these different situations should of course be kept as low as possible. If a large number of applications send reports at the same time, momentarily a larger amount of band-width will be used as the total number of messages increases linearly with the number of users. The obvious solution is that every application should not send its report at the same time as every other application, but instead utilize a
back-off and delay method based on the available bandwidth, the current number of members in the session and other reports received.

Reference [16] presents a mechanism for calculating the delay between control messages based on the mean size of the messages, the available bandwidth and how much of that bandwidth is reserved for control messages. The result of this mechanism is that the total amount of bandwidth used stays approximately constant independently of the size of the messages and the number of users in the session. This mechanism is reused in the manager framework (see Section 2.5 for more details about the delay calculation) with the difference that a larger portion of the bandwidth is allocated for the control and information messages to allow for faster interaction. Using this dynamically calculated delay, information reports can be sent to the whole group in a scalable way.

This reporting system can both be used for regular reports where applications report predefined information in regular intervals and for on-demand reports where applications report information based on requests. These two cases can also be combined where a manager requests applications to include some information that changes often in each regular report sent. This is for instance used for retrieving the amount of bandwidth currently being used by each member within their session.

To allow for administrator mobility all reports are always sent to the whole group using IP-multicast. This allows for different manager applications to be active within the same session at the same time without requiring that information is duplicated in the network (by different applications requesting the same information). This means that administrators can monitor and control their system from any host within the network.

### 2.2 Control

A pre-requisite is that the system should be able to control both single applications and groups of applications. Of course, the latter includes that if the same parameter is to be sent to all applications only one message should be sent and not one specific and identical message to each single instance. This leads to that messages should be sent using IP-multicast within the group just as the earlier discussed information reports.

To allow for easy messaging, a message and control protocol is needed. This control protocol have to support dynamic addressing of applications and parts of applications. Here we propose an agent based architecture where developers can easily reuse components and agents within different applications. An agent is a software component that resides within an application and is responsible for one specific task. For instance, a video-agent would be responsible for capturing and displaying video data and a database-agent would act as an interface to a database-engine. Numerous agents can and are normally deployed within one and the same application. Figure 1 shows some example agent based applications (especially note how the same agents are being reused in several different applications).

To support for dynamic addressing of applications and agents the Control Bus - CB [17] was chosen. The CB is designed for transmitting messages both within applications and between different instances of applications. The control bus is not tied into any specific transport mechanism but it is designed to be used together with any underlying reliable transport protocol.
The format of the CB messages are based on the message formats presented in [8] and [9]. Each message is completely text based and a message consists of four sections:

1. The **from section** which identifies the sender uniquely. The address field consists of four parts: Host / CB-id / Agent-type / Instance.

2. The **to section** which identifies the destination of the message. The address format is the same as for the from section with the exception that any part can be replaced with a wild-card. This allows for simple and dynamic addressing of messages where a message can be directed to a single agent, all applications within a session or a group of agents independently of which application they reside in.

3. The **message id section** which together with the from section forms a unique identifier within the session. This section consists of a sequence number that is increased for each transmitted message.

4. The **message section** which contains the message itself.

The traffic on the CB can be both one-way and two-ways, i.e., messages can be of the information-type where an agent is announcing something and is not expecting any response or it can be a request for more information where the requesting agent is expecting a response.

To simplify the protocol only text based arguments and values are allowed. If a specific application needs another type, then that application has to take care of conversion to and from a text representation. The use of a simple and pure textual message format makes it easier to create a platform independent messaging system. It also makes it easier to integrate the control bus into existing applications. As the target of this work is to create a framework that should support developers in the process of creating new distributed applications, the choice of using a text based protocol also makes it easier for developers to debug their applications and make them more stable.

Of course, there is a cost of using text based messages, both in processing and in the amount of bandwidth used but remember that the focus here is to support the transportation of small information reports and occasional control messages. Therefore, it is argued that the overhead in converting the messages between application specific formats and their
corresponding textual CB representation can be neglected. For an evaluation of the amount of bandwidth needed for control and management, see section 2.4 below.

2.2.1 Resource Discovery and Messaging

Until now, it has been discussed how to get information from applications/agents and how to address messages. We now continue with discussing the resource discovery and actual basic messages needed in the framework. There are two aspects of resource discovery, finding the actual agents to control and finding out what control messages they support.

An important aspect is to be able to find which agents are available within a certain group. Administrators should not have to keep track of which users use which applications and when. This process is supported by the alive message which requests all agents to return a short answer if they exist and the info-request describe message for retrieving a textual description of the agent. This allows the administrator to dynamically build a list over currently active agents.

In difference from SNMP we propose that every controllable agent should be able to supply information about which variables are controllable in that particular agent. This leads to a dynamic environment where every agent can evolve and be further supported with new control functionality as needed. The alternative would be to define all controllable variables and accessible methods in definition documents and very rarely change these. We argue that this creates a stale environment that puts to much demand on the design phase instead of allowing developers to extend the interface as needed and promoting an evolving environment. This discovery process is supported by the info-request describe-all message which requests a summary of all supported commands. Each command can be either write-enabled, read-enabled or both. The framework does not separate commands from get/set of variables in an agent. A command invocation is modeled as a set (with or without arguments).

Finally, the control and management framework must contain messages for actually invoking the get and set methods and these are called get-request and set-request.

Figure 2 show a number of example messages with corresponding control bus headers. Using these basic messages more information about the agent of interest and information about which methods it supports can easily be found.
2.3 Automatic Response Control Filters

It is not always that a human person can be available to monitor the activity within an organization. To target this, the framework contains the functionality to set up so called automatic control response filters. These filters can be set to respond to special conditions within an installation with predefined actions. For instance, if a user suddenly starts transmitting video with a bandwidth above a specific level a control message is sent to that application to lower the allowed bandwidth. These filters can also be set to trigger external programs which allows for easy extension of the control framework with installation specific applications.

2.4 Bandwidth Usage

Table 1 presents the number of messages and bandwidth consumed for doing a number of control tasks. Column 1: The task being performed (see below); 2: Total number of packets transmitted; 3: Number of bytes consumed without counting the underlying transport protocol; 4: Number of bytes consumed including the transport protocol using the reference implementation (see Section 4). Additionally, numbers for the effect of using unicast (top data in each row) versus multicast for transporting the control messages is presented. Packet loss and retransmissions are not considered as that is very network, situation and implementation specific.

<table>
<thead>
<tr>
<th>Task</th>
<th>Message Details</th>
<th>Unicast</th>
<th>Multicast</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1 - Resource discovery: find all video agents in the session. For the unicast case, it is assumed that the resource discovery is done manually and therefore not included here. Packets: n+1 Bytes excl.: 63<em>1 + n</em>92. The constants are based on the mean size of normal messages and is the number of agents in the session.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 2 - Information retrieval: get the current bandwidth setting for video transmission from all agents. Packets: (n + n) vs. (n + 1) Bytes excl.: (109<em>n vs. 80) + n</em>103</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 3 - Variable modification: set the bandwidth to be used by the video transmitter in each agent. Packets: n vs. 1 Bytes excl.: 113*n vs. 84</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 1. Number of packets and bytes needed for a number of different control tasks using unicast versus multicast in a session with 100 agents.

<table>
<thead>
<tr>
<th>Task</th>
<th>Number of Packets</th>
<th>Bytes excl. transport</th>
<th>Bytes incl. transport</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>12899</td>
</tr>
<tr>
<td></td>
<td>101</td>
<td>9263</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>21200</td>
<td>28400</td>
</tr>
<tr>
<td></td>
<td>101</td>
<td>10300</td>
<td>13936</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>11300</td>
<td>14900</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>84</td>
<td>120</td>
</tr>
</tbody>
</table>

It is hard to compare the amount of actually used bandwidth to other systems as it depends very much on the underlying transport mechanism used and therefore we limit ourselves just to compare the usage of unicast vs. multicast which is especially noteworthy. The bandwidth saving by using multicast instead of unicast for controlling agents is in task 2 of the ratio 1:2 and in task 3 as high as 1:n.

2.5 Delay

Due to the bandwidth control in the framework where agents calculate a delay before sending messages, there is an total delay before a task can be completed. This delay is dependent on several different parameters such as the number of agents in a session (collected from the underlying transport protocol if available otherwise based on earlier ‘alive’-tasks), the available bandwidth (usually depends on the IP-TTL/scope of the session) and the average size of the messages being sent including packet headers. The delay is calculated as shown below:

\[
\text{delay} = \left(\text{Rand}(0,1) \times 8 \times \text{size} \times n\right) / \left(0.5 \times \text{BW(TTL)} \times 1024\right)
\]

Table 2 presents the approximate delay for task 2 and 3 presented above. The network transport delay is not included here as that depends on the network setup and geographical conditions. The table shows that the delay gets long for large sessions with a large scope (e.g., about 8 minutes for 10000 agents in an global session). The reason for this is the very restrictive bandwidth allocation for global sessions in the current implementation. Note, that the exact amount of bandwidth to be used is implementation and session specific and depends on the underlying transport mechanism. For sessions with smaller scopes the delay is acceptable and usable even with up to 10000 agents (5-15 seconds to complete a message transfer).
2.6 Security and Privacy Issues

It is important that not just any user that have access to the control and management framework can control any other application. This is a privilege that should be reserved to authorized users.

Within the control and management framework this is currently solved using public/secret key based digital signatures [18]. Public/secret key technology means that each key is divided into a public and a secret key-part with the convenient property that the respective part cannot be calculated from the other. If a user encrypts something with the public part it can only be decrypted using the corresponding secret part. This can be used for digital signatures by calculating a digest (a unique digital summary of the data) and then encrypting it using the secret key. Anyone that have access to the public key can then decrypt the signature and compare the resulting digest to a digest that is calculated locally on the data in question. Note, that a correct signature can only be generated by using the correct secret key.

A predefined public key is stored in each client application and when a control message is received the optional accompanying digital signature is verified against the predefined public key. If the signature matches, the command is carried through and if it does not, a warning message is sent to the group warning about that there might be a vicious user trying to break the application control scheme. The use of digital signatures allows the messages to be sent in plain text and even if a message and its signature is snooped on the network, the snooper cannot do any harm within the session as s/he does not have the corresponding secret key needed to generate new signatures.

The predefined public key can be specified in the installation program for automatic handling within an organization. An optional override public key can also be specified and can be used to change the normal operational key if its corresponding secret has been compromised. This override key should of course be stored in a secure way and is only to be used in very rare cases.
The same digital signatures can be used for enabling privacy of reports if needed by encrypting the reported data using the public key. This means that only the users that have access to the corresponding secret key can decrypt the report and view its contents.

3. Usage Scenarios

This section presents a number of usage scenarios of how the management and control environment can be and is being used.

3.1 Session Membership Management

The mStar environment can be used for light-weight desktop conferencing. It allows any user that have access to the application to transmit audio and video within a session.

Using the control and management framework, administrators can get an overview of the members of a session and they can control who within the session should be able to transmit. This can be controlled in real-time on a user by user basis.

3.2 mWeb Configuration Control

When the mStar Web presentation system\(^1\) was deployed within a large organization (several hundred installations) and during a critical session broadcast a problem with the existing environment was found (due to the organizations special setup of its computers) where Web slides could not be displayed on users local computers. The software solution was simple and found quickly but the problem was then how to spread the modified software and the new settings to all listeners. If the control and management framework had been in place at that time it would only had been a question of sending out the correct message to all active instances of the application.

3.3 Better Perceived Quality of Service using Adaptive Applications

The framework can be used to control the maximum allowed bandwidth within a session when it is noticed that the total utilization of the involved networks gets above a specific level. Together with automatic filters the framework can be used for creating a better perceived Quality of Service environment where a control process allocates bandwidth to session as needed and control applications so they cannot utilize more network bandwidth than allowed. Earlier this have only been possible using static configurations and control by an operator. Now it can be controlled dynamically in real-time depending on other concurrent sessions. This creates a set of adaptive applications that better utilizes the available bandwidth and scales to a larger number of simultaneous users and sessions as over-utilization by a greedy user/application can be prevented.

Imagine a network setup with four hosts where host A, B and C are running multimedia applications and X is the controller. The total bandwidth is set to 400 Kbps for all sessions

\(^1\) A system called mWeb for distributing Web based presentations using IP-multicast [19].
on the network. Host A belong to session 1 and hosts B and C belong to session 2. Session 1 has 25% (max 100 Kbps) of the total bandwidth allocated to it. Initially only host A is active and it is transmitting with a bandwidth of 600 Kbps which is above the allowed maximum. The controller gets information about this via the regular reports and changes the transmit bandwidth of A to 400 (as nobody else is currently transmitting, A gets all available bandwidth allocated to it) Kbps via a control bus message. Next B and C joins and both start transmitting at the same time. The controller now lowers the bandwidth of A to 100 and sets the maximum allowed bandwidth for B and C to 150 Kbps each. After a short while the controller notices that host B is only transmitting with a bandwidth of 50 Kbps (the user selected to only transmit with this bandwidth) and therefore increases the available bandwidth for host C to 250 Kbps. When host A later leaves the session the controller reallocates the bandwidth from session 1 to session 2 and sets the maximum allowed bandwidth of C to 350 Kbps (note, that host B is still only transmitting with a bandwidth of 50 Kbps).

The framework can easily be used by applications to retrieve information about the currently allowed maximum bandwidth when the application starts. Instead of building this logic for adaptive applications into the applications themselves it can easier be controlled in real-time what policy should be used at a specific moment and installation.

### 3.4 Better Group Awareness

The mStar environment can be used as an electronic corridor where every user that wants\(^2\) transmits a low bit-rate video stream from their office. This allows for their colleagues to peak into their offices and get a view of what they are currently doing. This can be used for seeing if the other party is available or not before trying to contact him/her. It can also be used to create better awareness in a distributed group environment where users geographically separated (e.g., working in different offices or from their homes) can see if their colleagues are working as well. In the existing mStar system every user sends the amount of video they decide themselves, independently if anyone is actually viewing the video-stream or not.

Using the control framework the receiving mStar instance should instead notify the other party’s application when the local user is looking on a particular stream. When the other, remote application gets notified it should increase the amount of bandwidth used for that particular video stream. Further, the more users viewing the same video stream the higher bandwidth should be allocated to that stream.

The same concept can of course be used for real-time electronic meetings and lectures where the current number of viewers controls the relative bandwidth of a specific video stream. Note, that this example differs in its architecture from the earlier one in that it does

\(^2\) It is outside the scope of this paper to discuss why users would want to send video from their offices but experience have shown that camera shyness can be overcome after the added value to a distributed working environment is understood and that the purpose is \textit{not} for the boss to monitor the employees but to support the daily teamwork.
not have any manager process controlling the session but instead all control is handled within the session itself.

4. Implementation and Future Work
A reference implementation of the control and management framework have been implemented using the Java programming language and is currently deployed in several different products and applications that are available on the market.

A graphical front end and example control application called the multicast Manager - mManager have been designed and constructed. It allows a user to retrieve information about currently running applications and control these applications. For an application to be controllable by the mManager it has to contain the control bus software.

As the whole framework is IP-multicast based an mManager can be started on any host in the network and any number of simultaneous instances can be running at the same time. This allows for flexibility for the administrators as they will not be bound to any specific control station which is normally the case in large systems.

The underlying transport protocol for reliable multicast currently being used is SRRTP [4] which is an extension of the Real-time Transfer Protocol [16] for reliable transfer of data. It is based on the ideas from the SRM framework [20] with the key features of all members in a session helping out with repairs and heartbeats from senders together with negative acknowledgments are used to detect and signal packets loss.

The programmatic interface to the control bus is a simple interface where each agent registers which methods it wants to export. The registration includes a textual description of what the access method does, how many arguments it takes and which local methods should be called on get/set requests.

The security parts discussed earlier are also implemented using the Java standard libraries for key management and cryptography.

4.1 Future Work
An open user interface issue is how changes to an active instance of a program should be presented to the user and more precisely which changes are significant to notify the user about. For instance, if a controller requests an application to lower the bandwidth for its outgoing video stream by 1 Kbps. Should this cause the user to be notified or not? In this case it depends on the current bandwidth being used. In the current implementation of the framework it is left to the application to decide which changes should result in the user being notified.

The security system is still in its infancy and have to be further investigated. Specially the interface to the agent of how to specify who has access to which methods and information have to be further investigated and defined. This is a delicate problem as programmers want a simple but at the same time powerful interface.
The current automatic response filters are very simple and limited due to the lack of a good and simple action specification method. Initial interfaces of interest are SMAP [7] and the autonomous agent services of the Java Dynamic Management Kit [21]. Note, that the target is still to create an Java independent framework, i.e. one that is not dependent of the Java runtime environment.

With the framework in place it also would be interesting to pursue the issue of software updates by distributing software components using IP-multicast for dynamic on-demand update of software and permanent changes to software installations.

5. Conclusions and Summary

In the introduction we set out a number of requirements that the framework have to support. This section evaluates how these requirements have been addressed in the framework design.

- **The framework have to support large groups of applications, both for reporting and control.** This have been addressed by using IP-multicast for both control and reporting. Also, the delay between messages is dynamically calculated based on the current number of members in a session and the currently available bandwidth.

- **The framework have to support efficient control of groups of applications without the need to send control messages to each application explicitly.** The control bus allows single messages to be addressed to more than one application/agent.

- **The framework have to protect users from unauthorized control of their applications.** This have been addressed by including support for asymmetric cryptography and digital signatures.

- **The framework have to allow developers to easily insert control access points in their software.** This have only been addressed from a Java perspective where a simple but powerful API have been defined.

- **The framework should be as independent of the underlying software and transport technology as possible.** The whole framework is independent of the underlying transport mechanism but requires IP-multicast for scalability.

- **The framework should allow for scalable resource discovery of both available control objects and controllable variables and methods in these objects.** This have been addressed by allowing controllable agents to announce themselves and provide information about available control points.

This paper presents a framework for control and management of software applications. It allows applications to distribute messages in a scalable way, with regard to both the number of applications currently running and the available control bandwidth. This is done using IP-multicast, a messaging platform called the Control Bus (CB) and a reliable multicast protocol (SRRTP). Note, that the whole framework is designed without any special requirements from the underlying transport mechanism as long as it is transport reliable and uses IP-multicast (unicast can be used but the framework becomes much less scalable then).
The novel usage of IP-multicast for management and control creates a mobile and scalable framework that can be used for a number of different applications. We presented an evaluation of the bandwidth usage and the total delay for doing control tasks.

To utilize the control and management framework and administrate running applications, a program called the multicast Manager - mManager have been developed. The mManager gives the administrator an overview over currently running applications and their agents. It also allows administrators to control these applications.

We presented how the framework can be used for creating better group awareness in a distributed real-time environment. Further, it was presented how the environment can be used to create a better perceived Quality-of-Service environment for distributed media applications where bandwidth is dynamically allocated between active sessions and the total amount of used bandwidth is controlled automatically by a control process. These different usage scenarios show that the framework is flexible and can be used for a number of different applications.

The framework is currently being evaluated in the industry so it is too early to draw any final conclusions about its practical usability but initial tests show that it works and is powerful enough to be used in real deployed application in the industry (note, that the CB and the SRRTP have both been used in large scale earlier in other contexts and applications for several years and should therefore be considered as stable).

The work presented in this paper is based on requests from both academia and industry. We believe that this framework simplifies the administration and control of large groups of distributed applications and real-time multimedia sessions.

References


Acknowledgments
Thanks to Claes Ågren and Serge Lachapelle (Marratech AB), the people at Ericsson Data and the anonymous reviewers for comments, suggestions, and feedback.

This work is supported by the Centre for Distance-spanning Technology (CDT), Marratech AB and the Swedish National Board for Industrial and Technical Development, NUTEK.