2003:23

LULEÅ UNIVERSITY OF TECHNOLOGY

LICENTIATE THESIS

Application Semantics for Cost-Effective Media Distribution

STEFAN ELF

Department of Computer Science and Electrical Engineering
Division of Media Technology

2003:23 • ISSN: 1402 - 1757 • ISRN: LTU - LIC - - 03/23 - - SE

Application Semantics for Cost-Effective Media Distribution

Stefan Elf

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

JUNE 2003

Supervisor
Peter Parnes, Ph.D., Luleå University of Technology

Abstract

Public use of the Internet increases as wideband connections become pervasive and applications suitable for media distribution grow more popular. Group collaboration applications have likewise attracted interest during recent years. A growing attention towards wireless connectivity and business applications furthers a need for reliable communication protocols.

Multicast is a driving force for a wealth of new exciting applications involving one-to-many usage scenarios such as lecturing, or many-to-many scenarios, as in discussions and collaborative work. In some applications guaranteed delivery of every packet is not crucial, while in other this is a requirement. Error handling in reliable protocols can present a substantial challenge already in a homogeneous environment. Including a plethora of end-user terminal types with widely varying resources it becomes even more challenging. Protocols and applications must therefore be able to handle receiver and network link heterogeneity.

The work presented in this thesis aims to address some of the challenges facing the applications in this field. This is related to multicast and unicast alike and although the obstacles each of them must overcome may differ, there are similarities with regards to possible solutions with respect to error handling or resource allocation.

Error handling protocols are mainly proactive or reactive. Proactive protocols transmit redundant information along with the original information, enabling the receivers to repair lost packets without the necessity of feedback to the sender. Reactive protocols rely either on positive or negative feedback from the receivers in order to establish reliability.

According to a definition of semantic reliability the reliability concept can be interpreted in terms of application semantics. It is proposed in this thesis to view reliable multicast as a special case of semantically reliable multicast and to implement a dynamically configurable transport layer with an error-handling rule set that can be configured from the application or even from the sender in-session. It is further proposed to make use of the application's knowledge of specific semantics to improve on the recovery of lost packets, the handling of congestion, or intelligent sharing of resources.

Furthermore, this thesis presents a bandwidth-sharing scheme for group collaboration, essentially for video, which uses application semantics in the form of user hints. The presence of such events is made available to all potential senders via message passing between dedicated session members. As information relating to a user's interest in another user is conveyed, the sender may increase its use of resources on the expense of other senders. A scheme is proposed and a prototype implementation and experimental results are presented.

Contents

Publications

This thesis contains three publications that have been reviewed and published elsewhere and one that has been submitted for review. The introductory chapter provides a collected discussion on the issues covered in the papers, summarizes conclusions and suggests directions for future work.

Paper 1 S. Elf and P. Parnes, "A configurable transport layer as a cure for crying babies," in *Symposium on Computer Science and Electrical Engineering*. Luleå University of Technology, May 2001.

Paper 2 S. Elf and P. Parnes, "Applying semantic reliability concepts to multicast information messaging in wireless networks," in *IRMA Conference Proceedings: Issues & Trends of Information Technology Management in Contemporary Organizations*. Information Resources Management Association, May 2002, Idea Publishing Group.

Paper 3 J. Scholl, S. Elf, and P. Parnes, "Efficient workspaces through semantic reliability," in *10th International Conference on Telecommunications, ICT'2003*, February 2003.

Paper 4 S. Elf, J. Scholl, and P. Parnes, "Applying User-behavior to Bandwidth Adaptations in Collaborative Workspace Applications and Video Conferencing," submitted for review.

In addition to the above, the following paper has been submitted for review. This paper is not included in this thesis.

Paper J. Scholl, S. Elf, and P. Parnes, "User-interest driven video adaptation for collaborative workspace applications," submitted for review.

Acknowledgments

The work presented in this thesis could not have been done without the support and encouragement of a number of persons. I would like to thank my supervisor Dr. Peter Parnes for valuable discussions and for believing in that it is possible to pursue a research career besides an industry position. Thanks are also due to Dr. Arkady Zaslavsky at Monash University for valuable comments on some manuscripts and for discussions on research work in general. Thanks to fellow researchers within CDT for helping out by using the prototype implementations on Marratech Pro, thereby generating log data from experiments. Thanks are also extended to my fellow researchers at the institution in Skellefteå.

Thanks to Lars-Olof Martinsson, formerly Ericsson AB, for cheering on and for discussions on related subjects.

I have been co-operating with Jeremiah Scholl at CDT in Luleå. We have been working together in a constructive way in spite of us being located 130 km apart. To realize this co-operation we have been using email and Marratech Pro. In my view, our distance-spanning work has materialized the essence of CDT, Centre for Distance-spanning Technology, and this co-operation is greatly appreciated.

Most of my research has been funded by Ericsson Erisoft AB and later Ericsson AB, and the Centre for Distance-spanning Technology. Financial support from Mäkitalo Research Centre is gratefully acknowledged. Support during different periods was also provided by "Stiftelsen för Kunskaps- och Kompetensutveckling" - the Foundation for Knowledge- and Competence-development, the Swedish National Board for Industrial and Technical Development, NUTEK, and the VITAL project.

Last, but not least, my thanks goes to my family, to my partner Anna and our four children for their support and for, especially during the last few months, letting me disappear down to the "basement research office" as soon as I got home from my day-time job at Ericsson.

Skellefteå, June 2003
Stefan Elf

In memory of Ingegerd and Algot

1 Introduction and Summary

This licentiate thesis[1] presents a framework for the deployment of *application semantics* to cost-effective distribution of media. Media distribution can be made more effective by removing excess traffic caused by error correction, congestion, or sub-optimal resource allocation. To accomplish this, application semantics is used. Application semantics implies the use of the meaning of the information passed to an application, as viewed by the same application, or its user. In this work, user behaviour is also seen as application semantics, since the user's behaviour is tightly connected to the application semantics, for example when a user selects another user's video stream to watch, when a user mutes another user's audio etc.

The common denominator in this research is the use of application semantics to gain cost-efficiency in media distribution. This is then applied to (see figure 1.1 on page 2)

- error handling, with the special case of *crying babies* (paper 1)
- handling receiver and network heterogeneity by using a framework for semantic reliability (paper 2)
- application of the semantic reliability framework to e-meeting software (paper 3)
- user behaviour to control the resource allocation in e-meeting applications (paper 4)

The above bullets also reflect the structure of the remainder of this thesis introduction and summary.

1.1 Background

This work originates from Ericsson Erisoft's involvement in the Centre for Distance-spanning Technology (CDT) [1] at Luleå University of Technology (LTU). There were various reasons for Erisoft's involvement. One can for example point out similarities between a wireless setting including small hand-held devices, and an ordinary network with possible weak links, crying babies (see section 1.3.1 on page 5), network and receiver inhomogeneities.

[1]This thesis is for the degree of "teknologie licentiat", which is a Swedish degree between Master of Science and Doctor of Philosophy.

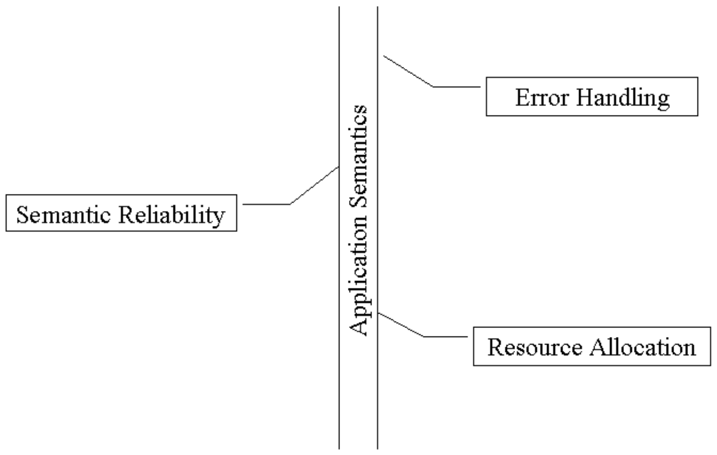

Figure 1.1: *The Application Semantics is the "red thread" of this work. Using application semantics as a base feature, we can apply it to error handling, to define a new kind of reliable protocols, and to handle resource allocation e.g. in video streams in a collaborative environment.*

Each of these aspects are discussed in the papers presented as part 2–5 corresponding to paper 1–4, and it will make references to the papers in an attempt to form a cohesive picture of the work carried out so far, as well as point out the path for continued efforts.

The most common way to work over the Internet has been over a point-to-point connection since both people and applications have interacted with each other on a node-to-node basis. A growing class of applications in which people collaborate has created a demand for something more efficient than simply broadcasting the data to all nodes in a network. This rather novel multicast technique turns out not to be very new. Multicast was first proposed some 20 years ago, but in spite of the increasing popularity of the applications it has until very recently been a rather limited facility. In the MBone [2], the multicast backbone, multicast capable routers are available and between these multicast islands, data is encapsulated, tunnelled, in unicast IP packets.

The work that is presented in this thesis has been motivated by the need for efficient error handling and resource allocation, especially in relation to the use of multicast, and especially in heterogeneous networks.

The base for the prototyping work and for on-line experiments has been Marratech Pro [3]. Marratech Pro is a commercial product that has been developed within CDT research work [4]. It is a proven concept sprung from various user scenarios that fits well with the kinds of situations that have been used to describe the problems to be solved.

1.2 Using application semantics

In language we talk of lexical rules, syntactic rules and semantic rules. While lexical rules teach us how to spell words and syntactic rules teach us how to put words together in sentences, the semantics deals with the meaning of what we have created with the first two rule sets. We can find parallels in media distribution where the lexical rules of data may represent each data packet and the syntactic rules may represent a sequence of packets dealing with some common task. The semantics of media distribution then becomes the meaning of the data information, as seen either by the application or, ultimately, by the user.

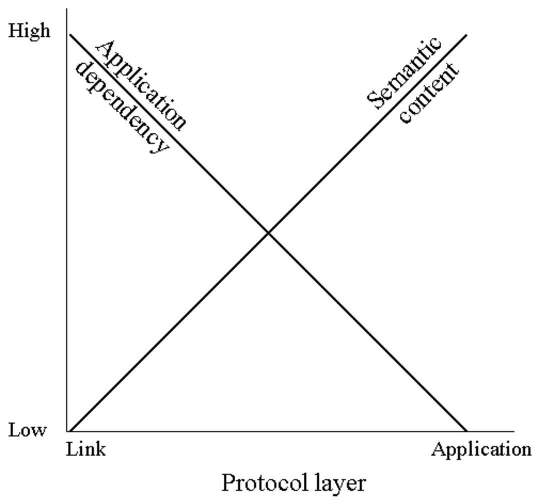

Figure 1.2: *The semantic content varies with the depth into the protocol stack. There is a trade-off between the semantic content available and the application dependency that is introduced.*

Looking at a protocol stack in general, the semantic content of different layers varies. The semantic content is greater closer to the application and lowers towards the physical layer. Figure 1.2 on page 3 shows this relationship in a schematic way. Of course, the application semantics exist also in the lower protocol layers, but these layers are not (should not be) aware of this. The figure also intends to show that implementing the handling of application semantics at lower layers introduces an application dependence, which is a drawback in some ways. It introduces a coupling between the layers that grows downwards in the stack. Thus, there is a trade-off between the amount of semantic content available at a certain protocol layer, and the application dependence that is introduced by pushing the application semantics handling down to that layer. It seems reasonable to assume that protocols, which explore application dependency to a high degree, i.e. protocols that are created for a specific scenario, will not be successful in the long run. The reason for this is that the applications and the scenarios shift, sometimes rapidly, over time.

Section 1.3.4 on page 7 discusses a proposal to integrate the use of application semantics with a lower layer handling, still requiring only a loose coupling towards the application, and thus no strict application dependency.

1.3 Error Handling in Reliable Multicast

This section discusses error handling in reliable multicast and relates to paper 1 and paper 2.

Reliable multicast is a prerequisite for a number of application types. Examples of these are shared whiteboards, on-line simulations, and on-line gaming (imagine being shot by the enemy in a game of Half-Life [5] without even knowing it — because of a few dropped packets; it gives a new dimension to the expression "before you know it").

Though there is an abundance of various multicast protocols, there are two large groups; sender-oriented and receiver oriented. Each of these carries drawbacks that the other one does not have. This is the reason for why a number of different, specialised protocols have been designed, each handling a dedicated situation, each making their own trade-off between advantages and drawbacks.

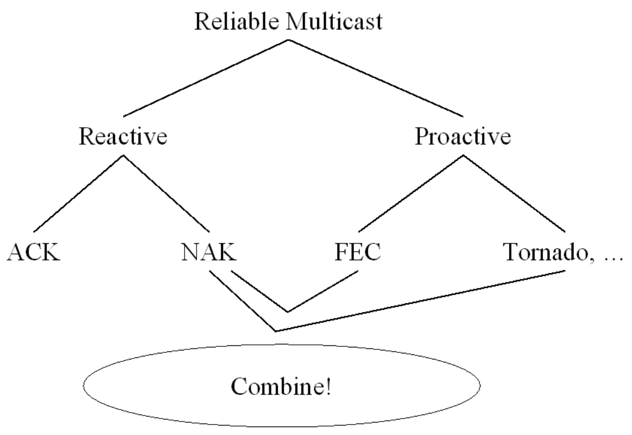

Figure 1.3: *Breakdown of multicast protocols with respect to error handling.*

Regarding error handling, there are also two major ways to handle lost packets. The methods are either *reactive* — they act to remedy a problem that has already occurred — or *proactive* — they seek to act so that the problem will never occur. A schematic breakdown of error handling in multicast protocols is shown in figure 1.3 on page 4.

1.3.1 Traditional Error Handling

The simplest reactive protocols will detect the loss of a packet and simply repeat it until its reception is acknowledged. Among the reactive protocols there are ACK-based (sender-initiated) as well as NAK-based (receiver-initiated). ACK-based protocols can be said to "poll" the receivers for their (the receivers') loss state by sending out a data packet. The opposite is true for the NAK-based protocols, where receivers that have detected a packet loss, will poll the sender for a repair packet.

There is a major problem with this approach — the ACK or NAK *implosion*. The problem exists for both strategies but is most pronounced for the ACK-based protocols. When there are a lot of receivers, there will eventually be generated so many ACK's that these will consume a substantial amount of bandwidth and possibly swamp the sender.

Using receiver-based protocols represents an attempt to remedy this, but when there are a lot of bad connections, there may instead be a NAK implosion.

To handle these drawbacks, a number of methods — each well suited for a special situation — have been devised. They range from building hierarchies of receivers in trees and assigning repair servers downstream to unload the senders' requirement for buffer space and CPU necessary for handling the receivers, to combinations of multicast and TCP/IP unicast. The common feature of these methods is that they try to partition the multicast tree in order to create many small problems instead of one big problem.

One special case that is relevant to inhomogeneous environments in wired networks, as well as in wireless environments, is the phenomenon named the "crying baby". A crying baby is thus a receiver that sits on a bad link, and who is constantly asking for resources to handle *its* problems. This crying for resources blocks other receivers from requesting repairs. One may note that even if the multicast tree is partitioned *ad infinitum* the crying baby will still end up in one branch, crying its eyes out. Since inhomogeneous networks in combination with disparate portable equipment will cater for more crying babies, there seems to be a need for a solution to this problem. The phenomenon and its relation to application semantics, and the notion if semantic reliability, is discussed in paper 1.

The other group of error-handling protocols in reliable multicast is the proactive. These use different means for prematurely sending out redundant information, which the receivers can use in order to recreate lost packets or parts of packets. By using slightly more bandwidth all the time, less overall bandwidth can be consumed, than by repeating lost information *en masse*.

1.3.2 A New Approach to Error Handling

How can application semantics make error handling more efficient? Consider, as one possible scenario, the presence of WCDMA cellular systems, or an abundance of radio-LAN (including IEEE 802.11x, HiPerLAN, etc.) transceivers, which provide ubiquitous access to the Internet within some defined area, exhibiting low cost and relatively high bandwidth. The deployment of interesting services directed towards a large number of cellular or mobile users at various events, or just being connected while on the move, requires that several issues be addressed. Some of these issues are

- security
- authentication
- reliability with error handling
- bandwidth allocation

Security and authentication issues have not been studied in this work. We assume that the semantics of these issues dictate that transmission must be traditionally reliable.

In a homogenous environment, error handling in reliable multicast can present a substantial challenge. When the scenario introduces a plethora of end-user terminal types with a widely varying capacity for reception, storage (e.g. for play-out buffers), etc, it becomes even more challenging. Among the most interesting questions regarding reliable multicast is how to avoid the strain put on network resources caused by the increased traffic, in turn caused by errors or, paradoxically, by congestion. By designing new, more efficient, protocols and multicast tree building and partitioning algorithms, it is possible to handle some of this excessive load. But a design is often a compromise where some factors are addressed, while others are neglected, dismissed, or at least not addressed to the full extent.

It is proposed that methods can be developed in which the context semantics of the data and the applications, and perhaps also the users, in a reliable multicast environment could be used to handle errors and decrease the strain on the network resources. By using such techniques, a better efficiency could be obtained.

It was considered important that the default behaviour is reliable multicast in the traditional sense. The semantics would then be an addition, a bump in the stack, which enhances the functionality of several current reliable multicast protocols.

1.3.3 Related Work

Other researchers have been working with semantics, but not all have been receiver-oriented and there has not been a collected view. The work that is closest to the present work was performed by Mauve and Hilt [6], by Pereira et al. [7], Rodrigues et al. [8], and Baldoni et al. [9]. Pereira et al. propose that as buffer occupancy reaches a high-water mark a constructed semantic in the packet header becomes active. Rodrigues et al. use message deadlines as a criteria and Baldoni et al. assign a life time to each packet. The packet can be purged when it reaches a deadline or its life time has expired. Mauve and Hilt presented a subscription-oriented API for reliable multicast.

ACK-based protocols, despite their inherent drawbacks for example relating to scalability, are now again being researched [10] in a small renaissance. One reason for this is probably the application of such protocols to wireless and cellular environments, where the number of participants in a session (connected to a base station, that is) seldom will grow very large and therefore their major drawback — poor scalability — would not present a large problem.

Table 1.1: *Example semantics and properties*

Semantic	Meaning	Property	Configured by
ABSOLUTE	All packets received	None	Sender/receiver
OBSOLESCENCE	Packets made obsolete	Seq.no.	Sender
REDUNDANCY	Redundant packet	seq.no.	Sender
LIFETIME	Packet life time	Time	Sender
EXPENDABLE	No crucial information	Boolean	Sender
DEADLINE	Latest arrival time	Time(stamp)	Receiver
CATCH-UP	Disregard packets	Seq.no.	Receiver
BEST-EFFORT	Fully relaxed	None	Receiver

1.3.4 Present Work

The approach presented in papers 1 and 2 in this thesis is a framework for application semantics with a set of semantic rules that can be dynamically configured by the application at any time, but that are carried out at a transport level. The primary rule-set is presented in table 1.1 on page 7.

Contrary to other approaches, this approach focuses on the receiver. The receiver can sometimes handle lost packets on its own, based on the local application semantics, or it can make use of information that comes embedded in the data. Each user of this framework only adapts itself and does not directly affect any other host. We have separated actions in the transport layer from judgements and decisions in the application parts of the framework. This allows for rapid handling in the transport layer. In the application layer, knowledge of application behaviour controls the way the semantic rules are set up. This should be understood in the context of there being a thin dividing line between the application and the transport layer.

The framework is schematically described in figure 1.4 on page 8. As is shown in this figure and as may be concluded from table 1.1 on page 7, semantic information can be either supplied by the sender or generated by applying context semantics in the receiver. When the information is distributed by the sender, there are drawbacks. First, the distribution of the semantic information implies knowledge of the receivers. Second, the information if sent, is not adapted to each receiver's capabilities and available resources. On the other hand, context semantics generated in the receiver does not have these drawbacks. By generating the information in the receiver and allowing this information to modify the same receiver's behaviour, has the advantage that each host will affect only its own behaviour. This is advantageous e.g. in the case of crying babies.

One could say that using context semantics is about how to strategically *loose* information without *missing* it.

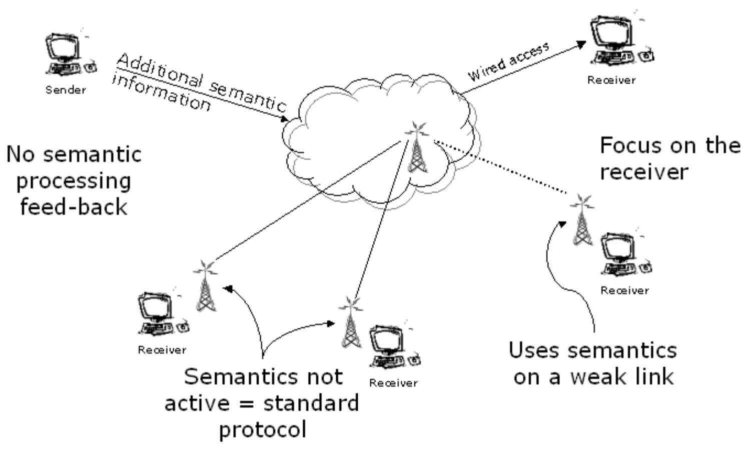
Sender
Additional semantic information
Wired access
Receiver
No semantic processing feed-back
Focus on the receiver
Receiver
Receiver
Receiver
Uses semantics on a weak link
Semantics not active = standard protocol
Receiver

Figure 1.4: *A schematic view of the framework.*

Error handling in general is discussed in a research report by Elf and Parnes [11]. The research report is not included in this thesis. General error handling is also discussed in the "related work" sections in paper 1 and 2 which also present the framework for error handling and semantic reliability summarised here.

1.4 Application Semantics in Collaborative Workspaces

This section discusses the application of context semantics to the collaborative workplace class of applications, and more specifically to the application Marratech Pro. Further below, a number of scenarios are discussed, which relate to paper 3 and paper 4 in this thesis.

Throughout history, people have always wanted to interact, to meet and discuss. Already since the advent of the telephone, we have been seeking to collaborate without regard to the sometimes-great distances between us, desiring to form groups based not on geographic location, but rather on interests and skills, moving the skills and ideas around rather than people. As electronic networks have emerged and grown, and bandwidth has increased applications that support voice over IP in various flavours are on the verge of replacing the landline telephone.

Body language is an integral part of how humans communicate. This is why voice-only or text-only communication cannot entirely replace real-life meetings. To cope with the shortcomings of the respective modes, people tend to for example either say "H" "I" in voice-only communication instead of laughing or using smileys in text-only communication, for lack of other means to express a mood that would otherwise be evident for the listener from the sender's body language.

The *collaborative workspace* class of applications offers a number of services such as voice, text-based chat, shared whiteboards, shared web browsing, and video communication, in order to create a virtual presence. Their key function is to satisfy the human need for not only hearing the words a person speaks, which is perhaps not even half of the message (according to some authors [12] as little as 30%), but also seeing this person. It also creates a *feeling of presence*.

New portable computing equipment is short of flooding the market. Cellular phones have already crossed the line between phone and palm-top computer. The challenges that media-distribution protocols must meet today have therefore largely become

- network and receiver heterogeneity
- scalability
- congestion in relation to TCP friendliness

Relating to group video conferencing, floor control has been applied in various flavours to implement bandwidth allocation schemes. Floor control has the drawback that it assumes a certain scenario only in which exactly one person is interesting enough for the community of viewers to be allocated significant bandwidth. According to our experiences, this one-type scenario is seldom present. To the contrary, scenarios tend to shift so that in one instance one single person is the most interesting sender, while in the next instance another person is the most interesting, and in the third instance, two different persons attract almost equal amounts of interest. According to our prototype experiments, mentioned in paper 4 (though not explicitly reported on in this paper), in a small group of about ten people, there can be as many as four interesting senders simultaneously, during a discussion scenario. If session members are also using the feature "video-follows-audio" that exists in Marratech Pro, the change of prioritised senders can be very rapid and for all practical purposes, one must conclude that this scenario also includes more than one prioritised sender.

1.4.1 Related Work

Previous research relating to the use of user behaviour include the work done by Kulju et al. [13] who investigated user hints in the context of video streaming, by Ott et al. [14] whose work focused on its use within their own 3D landscape. Amir et al. [15] also use the detection of user interest in order to allocate bandwidth in video conferencing. They describe the basic architectural components for schemes of this type.

1.4.2 Present Work

The present work addresses both efficiency and resource aspects. In paper 3 some scenarios applied to Marratech Pro are explored for the use of semantic reliability as a tool for increasing efficiency in collaborative workspaces. In paper 4 the allocation of video bandwidth in collaborative workspaces is specifically addressed.

The following section (1.5) discusses a few scenarios in detail, together with the proposed use of application semantics.

1.5 Usage Scenarios Applied to Marratech Pro

As a vehicle for testing the various applications of context semantics, Marratech Pro, a commercial product originally developed through research work at CDT, was used. Marratech Pro offers all these scenarios and also sports an highly experienced user group at CDT, in which people use Marratech Pro on a daily basis in the form of a virtual corridor. In the following sections a few scenarios are presented and the applicability of application semantics to them, is discussed.

1.5.1 Enhancing Whiteboard Efficiency

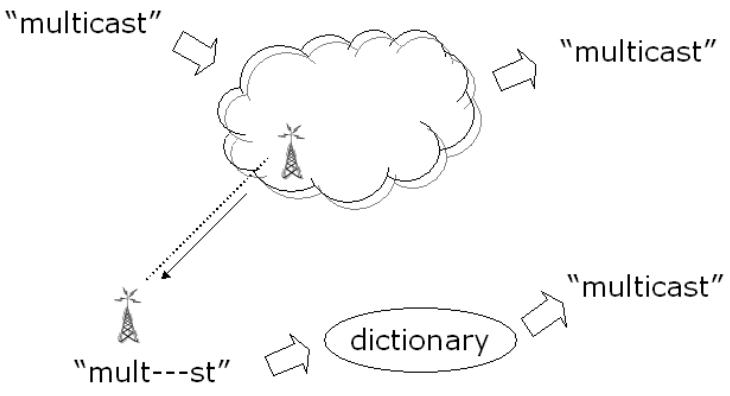

Figure 1.5: *Whiteboard text semantic reliability using local dictionary.*

This section presents a scenario, which is discussed in more detail in paper 3, which largely builds on paper 2.

Creating text nodes in Marratech Pro does not constitute a major problem as regards bandwidth consumption. On the other hand, it does exhibit a very large overhead and presents a highly instructive scenario related to explaining the advantages of using application semantics for error handling, in this case applied to reliable multicast, though there is not necessarily a restriction to multicast. See figure 1.5 on page 10.

In this environment, when a text node is entered into the whiteboard application, each letter is multicast to the other hosts that are members of the session. Each letter is sent in its own packet. If one or more letters of a word should be lost, a rather large overhead would result from the attempts to recover from these losses. Though each packet carries one letter, one byte of end-user information, each packet is 56 bytes long.

In this scenario, if the receiving application has access to semantic reliability, and when the application receives only some of the letters of the word, it uses a local dictionary to guess the current word. In doing so, the application will then adjust the semantic rule set to ignore these lost packets by not requesting them. If they are already requested, they are instead removed from the queue of outstanding packets. When the received packets are acknowledged, the application can optionally supply the guessed word, and the sender will at any convenient occasion, acknowledge that the guess was correct. Before this is done, it stands to reason that the guessed word must not be propagated to other session members.

The receiving application can thus use application semantics to recover from losses, and can also limit the amount of traffic on a link that may already be under stress

The technique is not limited to text nodes. Application semantics can be used to make entering sessions where a contents already is present in the whiteboard, more effective. Instead of updating all information on the present whiteboard page, rules for obsoleteness, life time, etc are applied and only valid data are processed.

A further whiteboard scenario is when a user draws a free-hand line in Marratech Pro. Members of the session would want to see the line appear as drawn. Not all pixels are multicast as the user draws, but instead every fourth pixel. These are reliably multicast to the session. If some of these packets were lost, it would be expensive to recreate them, especially since the whole free-hand line will be multicast as a bit-array, as soon as the user completes the object. If the receiving application can draw conclusions about the line shape during drawing, it could decide whether it is worth the effort to request repairs, or whether it can draw a good enough line with the available information.

1.5.2 Semantic Browsing

The scenario presented in this section is discussed in paper 3.

An interesting scenario with potentially larger bandwidth savings than the whiteboard text scenario is when a user leads a WWW browsing session. The sender, or leading session member, clicks on various links on web pages and all other users' Marratech Pro clients follows the leading user. All members can thus see the web page that the leader intended them to see. This can be followed-up by e.g. a discussion.

A user that sits on a weak link will obviously not have his or her web browser updated as quickly as other users'. Even if Marratech Pro carries facilities for ensuring that the picture is ready at all participants', there may be situations where the leader clicks on too fast for some clients. It is quite easy for a situation to arise, in which the currently loading web page is no longer relevant, since the leader has moved on to another web page. By using application semantics, the receiver can detect this condition, drop all packets related to the obsolete page and start receiving the current page instead.

1.5.3 Video Efficiency Through User Behaviour

The scenarios and approaches discussed in this section are discussed and analysed in paper 4. Paper 4 also presents the results from using a prototype with a simulator of some of the proposed features implemented.

Video is an important factor when creating a presence, as was concluded in section 1.4 relating to the importance of body language. On the other hand, video is still an expensive form of communication, in relation to bandwidth.

Different scenarios require different allocation schemes for video in relation to other media, such as text or web pages. In a common lecture scenario, one person is the main speaker. It is then straightforward to allocate most of the video bandwidth to that person and only rudimentary amounts of resources to other session members. I this way, it is ensured that all receivers can obtain the best possible view of the lecturer.

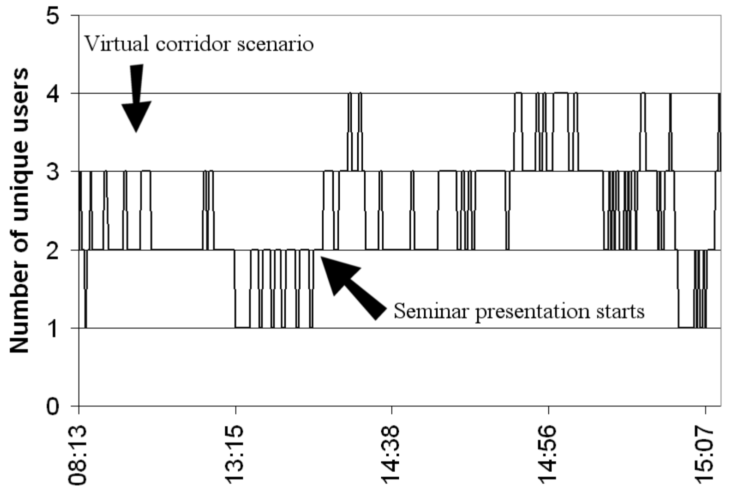

Figure 1.6: *The instant number of unique users considered interesting by other users.*

But scenarios change rapidly. In today's open-view society everything is discussed and soon the lecturer will not be the only person speaking, since the scenario has turned into a discussion scenario. According to the performed on-line experiments in a small user group consisting of nine members, occasionally as many as four different members were watched by others during a discussion scenario playout (see figure 1.6 on page 12).

1.5.4 Other possible scenarios

The usage scenarios presented in this section, are not thoroughly discussed in the papers but are presented here as they are relevant for further work.

In figure 1.7 on page 13 a scenario applicable to relaxed reliability is presented. A similar scenario was discussed in paper 2. The scenario relates to a portable computing device, such as the "fair navigator" mentioned in paper 2. If a disconnection occurs later, when connection is re-established, a lot of information will be "lost" for the portable device, although if analysed in view of application semantics, the information is not relevant any more. Using a relaxed reliability, this information need not be sent to the portable unit at all, thus conserving the limited bandwidth for better use.

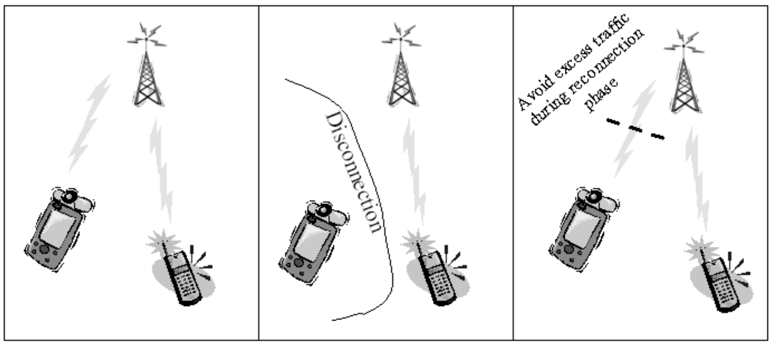

Figure 1.7: *Avoiding excess traffic at disconnection - reconnection of portable device, using semantic reliability.*

1.6 Implementation Issues

This section presents some implementation details related to the previously discussed issues. The architecture is discussed with regards to acquisition of semantic information and speed of computation, and efficiency.

1.6.1 Error Handling

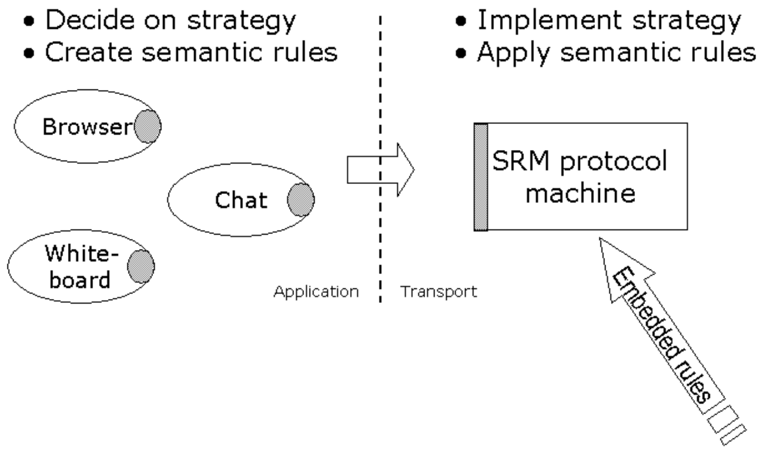

Figure 1.8: *Schematic architecture, especially relating to semantic error handling.*

A two-tier architecture is used as indicated in figure 1.8 on page 13. One part of the framework is located in the protocol stack. The other part is located in the application. In the transport layer we use a simple design for speed. A simple rule set enables us to minimise resources and not be too slow. In the application there is more time and information available. Here, the semantic properties of the application are known. Algorithms such as dictionary look-up can be used to decide which packets that the application possibly will not need, and that therefore can be discarded. In the application, we decide on a strategy and we use this for tuning of the transport layer part of the framework. Semantic strategy information can also be embedded in the incoming data, but the receiving application will still have the power to decide whether it shall be used or not.

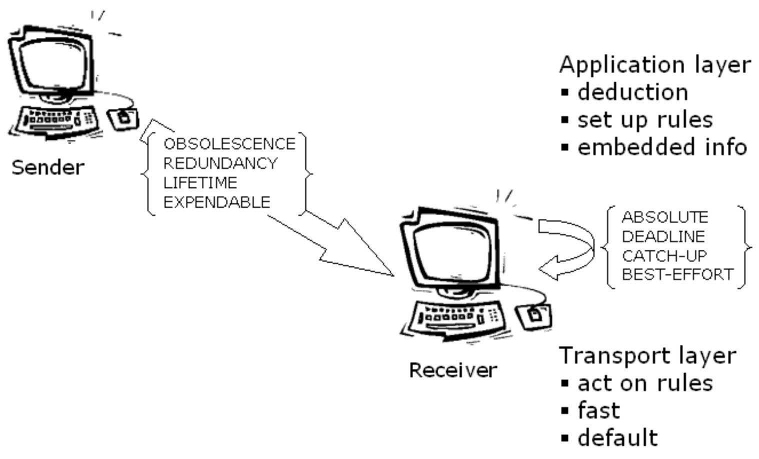

Figure 1.9: *Schematic implementation, especially relating to semantic error handling.*

A set of semantics are defined, examples of which can be found in table 1.1 on page 7 as can be seen in figure 1.9 on page 14. Even though these semantics can look like a scale of QoS levels with ABSOLUTE as fully reliable and BEST-EFFORT as not reliable at all it need not be viewed as such. The semantics used are simply ways of labelling the necessity of the information in view of the application.

In the transport layer the framework will sense whether any semantics are active. If none are, the behaviour defaults to the standard behaviour for the current protocol. If semantics are active the parameters are matched against the parameters of the currently analysed packet.

It is possible to discard packets without stopping the delivery of later packets to the application. It is also possible to scan the queues of outstanding packets in order to discard packets that have not yet been repaired, but where they for some reason are outdated or obsolete. The protocol has been modified to allow for the framework to set valid gaps in the packet sequence. Thereby repair requests for these packets can be suppressed.

The application must, according to its own semantics, decide whether the currently received packet allows for conclusion regarding subsequent, not yet received packets, for which a rule can be set up.

In the application part the specifics of each semantic are determined. The application creates a rule set in the transport layer, tied to the particular information stream of that application. In the current implementation, an IP address and a port number define such a stream.

Even if the resulting semantics are simple, the actions and decisions that must be made in order to arrive at them may be complicated. This is subject to further research. Obvious properties to analyse are time difference between packets and application time deadlines such as for overrun packets. Some information, like obsolescence will be embedded in the packets themselves, but it will be the application that decides on their use.

In order to tie this together, there must be a communication between the transport and the application parts of the framework. Often, there is only a very thin border between the transport and the application layer. The communication itself will not be complicated because of the simplicity of the semantic rule set construction.

1.6.2 Video Bandwidth Allocation

The present prototype implementation related to the allocation of bandwidth according to user behaviour is based on a number of "hooks" that are strategically inserted into the Marratech Pro code. It is all implemented in Java. Besides these "hooks", most of the implementation is located in a separate package, which makes the new code easy to maintain.

Since the design requires that receivers send messages to senders, declaring their respective interest in the senders, there must be a messaging system. The chat application in Marratech Pro is based on the so-called Control Bus [16]. The current implementation hooks into the chat mechanism though all messages are hidden from the users.

All prototype implementations can log information about users' actions. Still, one client is a dedicated logger and using the new `InChatMessage` class all receivers will send the logging information to that particular user.

1.7 Organisation of the Thesis

This thesis introduction consists of four parts each containing one paper that has either been reviewed and published elsewhere, or submitted for review.

All papers are reproduced here in their original form with the following exceptions. The numbering of sections, equations, figures, etc, has been changed to allow a common numbering scheme throughout the thesis. The bibliography for each paper is retained and there is no common bibliography for the whole thesis. Moreover, cosmetic changes have been made to figures and tables to make them fit the format of this thesis.

Part 2: The first paper, entitled "A configurable transport layer as a cure for crying babies", presents a case of special interest in relation to wireless connectivity where essentially one receiver on a lossy link eventually will consume most of the error handling resources, proposes to view reliable multicast as a special case of semantically reliable multicast, and to make use of the application's knowledge of its specific semantics to allow the application to relax the reliability requirement at the transport level.

Part 3: The second paper, entitled "Applying Semantic Reliability Concepts to Multicast Information Messaging in Wireless Networks", presents a framework for an application

- transport layer communication to implement a dynamically configurable transport layer protocol whose error handling rule set can be configured from the application or even from the sender in-session.

Part 4: The third paper, entitled “Semantic-Reliability for IP-Multicast Collaborative Workspaces”, examines how a specific collaborative workspace application can benefit from a general framework for classifying data based on semantic reliability.

Part 5: The fourth and last paper, entitled “Applying User-behavior to Bandwidth Adaptations in Collaborative Workspace Applications and Video Conferencing”, is an expanded version of a paper that is not included in the thesis but which is mentioned in the publications list, including a formalised bandwidth algorithm expression and extended analyses of the measurements taken from use of the prototype implementation. Furthermore, the algorithm has been implemented in the log data analyser and used for bandwidth simulations.

1.7.1 Contribution

This section explains the extent of my own contribution to the respective papers.

Paper 1. I am the main author of this paper.

Paper 2. I am the main author of this paper.

Paper 3. Jeremiah Scholl is the main author of this paper. The paper builds on the framework presented in paper 2 and on common ideas. I contributed part of the introduction, most of the related work, part of the discussion on obsolete data in web browsers and whiteboards, part of the whiteboard semantic reliability especially on the local dictionary, the figures, about half the summary, and a part of the conclusions.

Paper 4. I am the main author of this paper. I wrote most of the text. I was the primary contributor to creating the prototype and analysing the data from the empirical study. I formalised the bandwidth allocation algorithm, implemented it and tested the algorithm on the data acquired from the empirical study.

1.8 Summary and Conclusions

This thesis presents an approach by which applications can use context semantics to handle various situations ranging from error handling to resource allocation. By splitting the architecture in two tiers, the lower level tier can be made more general, less application and protocol dependent, while the higher level deals directly with the application semantics and configures the parameters for the lower layer.

By letting the scheme default to present standard behaviour, it is made probable that deployment of this kind of framework can be done in a few hosts at a time and thus, there is no need for a sudden change of a large number of hosts or protocols.

Reliable multicast implies that no packet must be lost. Packets are lost more often on weak links and when a packet is lost, the protocol will generate more traffic in order to correct the error. This can obviously be counter-productive. This thesis has presented a way to decrease this excess traffic by using that application's knowledge of which corrections that are actually necessary.

Application semantics can also aid applications in performing better by avoiding senseless updates and non-relevant dissemination of information. This pertains e.g. to the effective browsing.

Using application semantics and associated user behaviour for distribution of video bandwidth we can obtain a conservation of bandwidth according to certain constraints while preserving an adequate perception of the most important senders in a video conferencing session. This is the bandwidth algorithm description.

The conclusions made so far in the various papers that make up the thesis are, related to error handling

- Reliability is a relative notion and could most favourable be seen in relation to application semantics.
- A two-tier architecture will divide the computing burden between the application and lower protocol layers and will decouple the application from the protocol handling, thus making the protocol extensions less application dependent.
- Though semantic reliability has been researched by others, there has not been available a framework that puts application semantics into a larger perspective.
- Since each receiver only configures itself, dynamic changes in behaviour will, at least not directly, affect any other receiver. This can be useful for handling weak links such as in wireless connectivity.
- Semantic reliability can be applied also to make applications in collaborative workspaces more efficient by relaxing error handling in a fashion that the user can tolerate.
- Even though applications in a collaborative workspace have different properties, the semantic reliability framework is flexible enough to be applicable to several or all of them.

...and related to the use of application semantics and user behaviour in resource allocation

- In order to cost-effectively media streams under varying usage scenarios and in heterogeneous environments, applications must take application semantics, including user behaviour into account.
- Allocating resources according to our scheme, first giving minimal resources to all, and then allocating further resources according to receiving users' behaviour, helps in quickly establishing users' presence.

- By dynamically re-configuring applications based on the ultimate application semantics, the user behaviour, the resource allocation scheme will not be restricted to any single scenario. This is important since a usage scenario is seldom static.
- The proposed messaging system would not generate a significant amount of overhead, according to (limited) on-line experiments in an experienced user community.

My own main contribution that encompasses all the presented work is the framework for the use of application semantics in the different presented scenarios, that it to a large extent is receiver-oriented and that each receiver dynamically can re-configure itself in order to adapt to the situation at hand, be it regarding network conditions or user behaviour, and that the behaviour defaults to present standard behaviour, these extensions thus acting as an enhancer rather than a prerequisite, which will aid its implementation over different applications and networks.

1.8.1 Future Work

Originally, plans were to implement error-handling schemes in a network simulator. Since then, implementations have been mad in Marratech Pro. Though this is not a simulation environment, it has been shown feasible to implement prototype modifications according to the presented work, and extracting log information from Marratech Pro. Furthermore, this software is used at LTU on a daily basis and therefore presents a plausible way to also gain real-life testing on prototypes.

It seems likely that future work both with regard to error handling and with regard to resource allocation, would involve further implementation in Marratech Pro.

First on the agenda is running a full-scale user test deploying user behaviour controlled bandwidth allocation as well as semantic reliability for error handling. It will study how the prototype modifications work in the interaction with the users in relation to the perception of quality and presence. It will also study how network performance is affected.

1.9 Acronyms

ACK	Acknowledgement
API	Application Programming Interface
CPU	Central Processing Unit
FEC	Forward Error Correction
IEEE	The Institute of Electronics and Electrical Engineers, Inc.
HiPerLAN	High Performance LAN (European WLAN standard)
IP	Internet Protocol
LAN	Local Area Network
NAK	Negative Acknowledgement
TCP	Transmission Control Protocol
WCDMA	Wide-band Code Division Multiple Access
WWW	World Wide Web

Bibliography

[1] “The Centre for Distance-spanning Technology,” URL[2], Visited 2003-04-27.

[2] H. Eriksson, “Mbone: The multicast backbone,” *Communications of the ACM*, vol. 8, pp. 54–60, 1994.

[3] Marratech, “- The e-meeting company,” URL[3], October 2001, Visited September 9th, 2002.

[4] P. Parnes, *The mStar Environment - Scalable Distributed Teamwork using IP Multicast*, Luleå University of Technology, December 1999, Doctoral Thesis, ISSN 1402-1544 / ISRN LTU-DT–99/31–SE / NR 1999:31.

[5] Valve Software, “HλLF-LIFE, Award-winning computer game.,” URL[4], Visited 2003-04-27.

[6] M Mauve and V Hilt, “An application developer’s perspective on reliable multicast for distributed interactive media,” *Computer-Communication-Review. vol.30, no.3; July 2000; p.28-38*, 2000.

[7] J Pereira, L Rodrigues, and R Oliveira, “Semantically reliable multicast protocols,” *Proceedings of 19th IEEE Symposium on Reliable Distributed Systems*, pp. 60–9, 2000.

[8] R. Rodrigues, R. Baldoni, E. Anceaume, and M. Raynal, “Deadline-constrained causal order,” *The 3rd IEEE International Symposium on Object-oriented Real-time distributed Computing*, Mar. 2000.

[9] R. Baldoni, R. Prakash, M. Raynal, and M. Singhal, “Efficient Δ-causal broadcasting,” *Journal of Computer System Science and Engineering*, 1998.

[10] Byung Won On, Haesun Shin, Miae Choi, and Myong Soon Park, “A hierarchical ack-based protocol for reliable multicast in mobile networks,” *Proceedings of IEEE ICON International Conference on Networks*, pp. 359–62, 2000.

[2] <http://www.cdt.luth.se/>
[3] <http://www.marratech.com/>
[4] <http://www.half-life.com/>

[11] S. Elf and P. Parnes, "A literature review of recent developments in reliable multicast error handling," Technical report, Luleå University of Technlogy, Jan. 2001.

[12] Jian Wang and Donald G. Frank, "Cross-cultural communication: Implications for effective information services in academic libraries," *portal: Libraries and the Academy*, vol. 2, no. 2, pp. 207–216, April 2002.

[13] William Kulju and Hanan Lutfiyya, "Design and implementation of an application layer protocol for reducing udp traffic based on user hints and policies," in *5th IFIP/IEEE International Conference on Management of Multimedia Networks and Services, MMNS*, 2002.

[14] Maximilian Ott, Georg Michelitsch, Daniel Reininger, and Girish Welling, "An architecture for adaptive qos and its application to multimedia systems design," *Special Issue of Computer Communications on Guiding Quality of Service into Distributed Systems*, 1997.

[15] Elan Amir, Steven McCanne, and Randy H. Katz, "Receiver-driven bandwidth adaptation for light-weight sessions," in *ACM Multimedia*, 1997, pp. 415–426.

[16] P. Parnes, K. Synnes, and D. Schefström, "A Framework for Management and Control of Distributed Applications using Agents and IP-multicast," in *Proceedings of the 18:th IEEE INFOCOM Conference (INFOCOM'99)*, New York, USA, March 1999.

Part 2

A Configurable Transport Layer as a Cure for Crying Babies

A Configurable Transport Layer as a Cure for Crying Babies

Stefan Elf and Peter Parnes
Luleå University of Technology
Centre for Distance-spanning Technology
Department of Computer Science
SE-971 87 Luleå, Sweden
{self, peppar}@cdt.luth.se

Abstract

Group collaboration and media distribution applications have attracted more interest during recent years. With an increasing attention to wireless environments and business applications there is a growing need for reliable multicast communication protocols.

There have been mainly two approaches, proactive or reactive, to error handling in reliable multicast. Proactive protocols transmit redundant information along with the original information, enabling the receivers to repair lost packets without the necessity of feedback to the sender. Reactive protocols rely either on positive or negative feedback from the receivers.

A case of special interest in relation to wireless connectivity is where essentially one receiver on a lossy link eventually will consume most of the error handling resources. This problem has been addressed by some researchers, e.g. by the use of probabilistic methods.

We propose to view reliable multicast as a special case of semantically reliable multicast. We propose to make use of the application's knowledge of its specific semantics to improve e.g. on the problem with single, lossy, links. We suggest the design of a semantic rule set for the transport layer protocol, which can be configured by the application. Using application semantics would allow the application to relax the reliability requirement at the transport level.

2.1 Introduction

It is not until very recently that the Internet has been available to a general public. As more homes are connected by cable or stable radio links, Internet business cases become attractive. Where there is business we need security and security requires reliability.

As a result of people being commonly connected, there is a growing interest for establishing a mobile computing platform, enabling mobile connectivity. Parallel to a growing wired network, we will therefore also see a growing wireless infrastructure with all its implications. In this new setting the location of hosts will not be static and links will be established and broken depending on the user's location changes and the possible shortcomings of the wireless infrastructure. As the computing platforms shrink in size, users will become more mobile and then expect to be able to stay on line while on the move. Links to wireless hosts will have a much less stable bandwidth than we have been used to from wired links.

Capabilities for reliability are mainly implemented in the transport layer. A number of reliable multicast protocols have been proposed over the years, each adapted towards a specific problem area. A consequence of the conflict between the need for a collected view of reliable multicast services and the very different requirements and possibilities offered by various applications, is drawn in the creation of the Application Level Framing principle, proposed by Floyd et al. [1]. ALF has consequently grown in popularity during recent years, shown e.g. by Chawathe et al. in the RMX protocol [2], which operates in the transport layer as well as in the application layer.

Making a multicast protocol reliable is one challenge. But at the same time, the protocol must work efficiently not only for two users, but for hundreds or thousands, depending on the application. The bandwidth can be exhausted by re-transmissions as well as by control message implosions. Single, failing, receivers on lossy links, must not disrupt the session for other, more stable, hosts, by consuming most of the resources for error correction. This latter phenomenon has been described as the *crying baby problem* [3] and it is even more pronounced in a scenario with a mix of wired and wireless hosts.

Various methods have been used in order to lighten the load from the sender or to share the load among the receivers, depending on whether a sender or receiver based strategy has been chosen. It would seem that, at least in scenarios where the number of receivers is fairly high, the tree-based receiver oriented approach, has been the most successful, both in terms of sharing load, improving on locality of repair, and on overall protocol efficiency.

In a tree-based protocol, the remedy to an inhomogeneous group is to split the group. But if the group contains one crying baby, splitting the group will not help, since the receiver on the lossy link will obviously belong to one of the resulting groups. There have been probabilistic approaches to load sharing and scalability, which also lead to a remedy to this problem, such as proposed by Xiao and Birman [4].

Application level framing obviously will enable error handling according to application specific semantics. This will allow for error concealment strategies where the transport layer would simply ignore lost packets guided by suitable constraints and the application would conceal the lost information.

By allowing for the application to configure the transport layer protocol, we propose to use the best of both worlds, namely to combine the knowledge of the semantics from the application layer with the speed of processing of the transport layer. This will allow us to create a service that is “semantically reliable”, where the reliability will be put in the context of application semantics. We need not recreate or request a repair of a packet that is not necessary for the application to fulfil its task.

The remainder of the paper is structured as follows. Current error handling methods are reviewed in section 2.2. Next, in section 2.3 we propose and discuss a framework for a configurable transport layer protocol. The concept of semantic reliability is expanded on in section 2.4. Section 2.5 presents related work. Lastly, in section 2.6 we conclude.

2.2 Multicast Error Handling

In this section, the current methods of error handling in reliable multicast protocols will be reviewed.

Essentially, reliable multicast protocols are all either mainly reactive or proactive with regards to the way packet loss is handled. Reactive protocols will repeat information on detection of losses, while proactive protocols will use redundancy, or parity, information to supply the receivers with knowledge to repair losses concurrently to the original data transfer. The reactive protocols are either ACK-based or NAK-based. This simply implies where the responsibility for detection of packet loss lies. ACK-based protocols leaves this responsibility with the sender, while NAK-based protocols share the responsibility among all the receivers.

ACK-based protocols have an advantage in that the sender will always know when a packet can be discarded and the buffer space reclaimed. On the other hand, they suffer from the drawback that non-ACK'ed packets must be buffered by the sender and that the sender must keep state information for each receiver. Therefore this solution does not scale as the number of receivers grows. The amount of resources demanded by the sender will eventually become too high, or the number of ACK's sent by the set of receivers will exhaust the network or sender host resources and create an *ACK implosion*.

NAK-based protocols have the advantage of being scalable to a large audience. This is possible while each receiver is responsible only for its own state, but if there is a large number of receivers who experience packet loss of a magnitude, the sender will instead face a possible *NAK implosion*.

To avoid the implosion effect and to boost efficiency, a tree structure is often used. The tree structure enables reduction of the number of control messages and caters for a locality of repair.

Packet loss is detected by the use of timers. A NAK is sent when the timer expires. The expiry time is can be a function of the receiver's distance from the sender, or a random value. The Scalable Reliable Multicast [5] uses a combination of these methods to select the timer value. The receivers that did not send a NAK, will reset their own timers when then sense the NAK multicast by the "winning" receiver. Of course, repair data will also be multicast to the whole group.

Repeating packets either on loss detection at the sender or on request from the receivers generates excessive traffic that increases with the amount of loss. If the cause for the loss is congestion, this would seem to be somewhat counter-productive. Alternate proactive schemes have thus been explored. A proactive protocol is not dependent on an ACK or a NAK if the packet loss is within reasonable limits. Redundant information is incorporated in the packet stream. This allows the receivers to repair a certain amount of packet loss without feedback to the sender. The methods used include erasure codes and forward error correction schemes.

The Digital fountain approach using a special application of efficient erasure codes called *tornado codes* is discussed by Byers et al. [6]. The Digital Fountain can be especially effective in e.g. distribution of predetermined amounts of information, since there is no need for feedback at all. A receiver that looses a packet will just have to wait a while longer in order to obtain *any* few more packets to be able to reconstruct the missing ones.

The drawback carried by sending redundant information is obviously that the required bandwidth is increased whether it is needed or not. The data must also be blocked, and this is not always suitable for streaming media. The blocking will also introduce delays in the data delivery.

Since the most viable solution seems to be the NAK-based strategy, several researchers have aimed at improving on the drawbacks of NAK-based reliable multicast. The NAK implosion problem can be handled by using or creating a favourable topology. Among the receivers some can be dedicated to error recovery and thus the receiver NAK's need be sent only to these nodes using TTL scoping. The same nodes are responsible for repair actions and the NAK's never reach the original sender.

Not one single technique will prove to solve all the complex problems of reliable multicast. Hybrid protocols for reliable multicast have used combinations of different error correcting schemes, such as combining repeating protocols with redundancy. In RMX, Chawathe et al. [2] takes this a step further by combining multicast with unicast TCP traffic.

It would seem that the crying baby problem is partly addressed by tree-based systems, by forward error correction methods, and by probabilistic load-sharing systems.

2.3 A cure for crying babies

Handling crying babies without loading other hosts in the network requires that traffic to other receivers not be impaired and that proxies are not overloaded.

In order to fulfil this, there must be a low amount of feedback information generated per detected loss and servicing host. This can be accomplished by e.g. splitting groups to lessen the probability of there being a crying baby in the group, or by enabling hosts to share the load among themselves. This seems not to be a widely addressed problem.

In this section, we propose a method for error handling especially in inhomogeneous environments in reliable multicast protocols, which should be possible to implement on top of some existing multicast transport protocol.

When applying ALF principles, it is understood that the error handling should be done in the application. The advantage with this is that the application knows the semantics of the application data, which are essentially unknown to the transport layer where the information appears as a packet payload. The drawback is that handling errors in the application can consume considerable resources such as buffer space and CPU power. If we are considering a wireless scenario also, this can be constraining properties.

One way to handle this problem is to be efficient in the application. There is always a tradeoff between having access to more information and being able to process faster, as you move down a protocol stack. We speculate that an optimal way to proceed would be to define, based on received data and knowledge of application semantics, an error handling rule set in the transport layer, which can be configured by the application to handle certain packets in a specific way. Therefore, we propose to create a configurable transport layer protocol based on the notion of semantic reliability 2.4.

By making the transport layer protocol configurable, the transport layer will probably be differently configured at each host. The configurable parts of the transport layer protocol will not be engaged in the interactions with other hosts. They will instead serve the local application with timely decisions regarding lost packets. Since all hosts can have different semantic configurations, this implies that a crying baby need not disrupt the session for other participants.

The configuration of the transport layer could either be done dynamically, by feedback from the application, or during session start-up having a sender start the session by uploading transport layer configuration parameters for the session. These are recognised by the transport layer directly as a dedicated packet type. The parameters can also be recognised by the application and returned by the application to the transport layer.

Receivers who join a session late and wish to update their session configuration can do so by multicasting a dedicated message to its neighbours. The session configuration is then multicast to the same scope. It is to be noted that receivers who do not have the session parameters, can still receive from and transmit to the group. This configuration would not be a prerequisite, but rather an enhancement that can be used with any protocol.

One example of a session configuration is the implementation of a deadline constraint, which can be described by the following scenario.

In an audio application it is not essential that all packets be delivered to the application. In fact, if the audio application has lost a packet, it can sometimes simply replay the last received packet instead of the lost one, without any serious deterioration to the output signal. But the transport layer would not know of this, and if there were packet losses, the transport layer would request lost packets, even if the application could decide that they were not necessary.

According to our proposal, the application would be able to configure the “deadline” semantic of the transport layer. If, at time t_0 the packet with a time stamp t_p was to be played out by the application, and if a packet must be placed in the playout buffer a time t_l before it is going to be played out, the application can configure the constraints $t_0 - t_p$ and t_l in the transport layer protocol. Each time a packet is received in the current stream, the transport layer would decide if for the current packet $t - (t_0 - t_p) - t_l < 0$. Were that the case, the packet would simply be discarded. The configuration parameters can preferably be updated by the application during run-time.

In a shared white-board application most of the time it is imperative that packets are not lost - but not always! Imagine that a user draws a circle and another user does not receive the circle-packet. But before this error can be corrected, the first user deletes the circle. Now, there is no meaning in re-sending the original circle-packet to receiving user. The sender can obviously detect this situation, and remove the original circle-packet from its buffer. Prior to sending this packet, it is marked as rendering the circle-packet obsolete and redundant. The receiver who is still missing both packets can, upon reception of the circle-delete-packet, remove the request for the originally lost packet, and drop the circle-delete-packet. Neither packets need ever reach the application.

This behaviour is the result of use of application specific semantics, which are downloaded to a configurable transport layer protocol at the start of the session.

2.4 Semantic Reliability

This proposal is based on the concept of semantic reliability, which has been exemplified above. If we accept that a reliable transmission must not necessary imply that all packets reach their destination, but that the notion of reliability must be put in the context of the application semantics, we can begin manipulating with the receiving queues by adjusting the behaviour of the transport layer protocol.

The configuration information that is either distributed in a dedicated packet type, or downloaded from the application, contains a meta-semantic which describes the type of semantic to be used, and possible parameters. Which kind of semantic this would specifically be, depends on the application in question. A few examples (obviously, some of these have already been explored by other authors, as described in the following section) follow

- **ABSOLUTE** Every single packet must be received. Configured by a flag.
- **OBSOLESCENCE** The sending application will provide information as to which packets the current packet will render obsolete. Configured by listing packet sequence numbers.
- **REDUNDANCY** The sending application provides information as to whether this information will make older information redundant. Configured by listing packet sequence numbers.
- **LIFETIME** The sending application supplies a measure of life time for each packet. Configured by a time constraint value.
- **EXPENDABLE** The packets that are marked expendable by the sending application can be dropped without hazard for the application semantics. The receiving application can also provide this information. Configured by a flag.
- **DEADLINE** If the packet arrives after the provided deadline, it must be discarded. This kind of deadline can be provided either by the sender or by the receiver. Configured by a time limit value.

More semantic measures can be included depending on the application. For each semantic, there will also be a set of parameters provided in the configuration, which will be used by the transport layer protocol.

2.5 Related work

In section 2.1 a brief survey of the closest related work is given. According to this, errors are handled either in the transport layer, where they are subject to re-sending original data upon detection of packet loss. There are various techniques to reduce the number of control messages by using hierarchic models or by multicasting control information and/or repair information. The RRMP proposed by Xiao and Birman [4] can also handle inhomogeneous environments, since it shares the burden among all receivers in a local group by using a probabilistic approach. This work is related to our proposal in the sense that it also solves the crying baby problem that this paper addresses.

The work presented by Pereira et al. [7], Rodrigues et al. [8], and Baldoni et al. [9], comes closest to our proposal.

Pereira et al. propose that as buffer occupancy reaches a high-water mark, a constructed semantic in the packet header becomes active. Each packet header contains information pertaining to which earlier packets it renders obsolete. This can be used to purge packets from receiver buffers. It could well be used to clean up sender or repair server buffers. Also, during congestion, receivers may drop all packets that do not render any packet already in the buffer, obsolete. During normal, i.e. not congested, operation, this mechanism is not active.

This kind of solution is generic with regards to the receiving end. At the sender, though, the construction of the obsolescence information if clearly application dependent.

Rodrigues et al. use message deadlines as a criteria and Baldoni et al. assign a lifetime to each packet. When a packet reaches a deadline or the lifetime has expired, it can be purged, possibly to be replaced by a subsequent packet.

These obsolescence techniques and the efficiency of them, all depend on application semantics. They also rely on the application to provide the semantic to the transport layer.

Our proposal differs from the referenced work, in that we aim to create a configurable transport layer protocol in which the sender side application can feed information to the transport layer, which can be interpreted at the receiver side, and in which the receiving application can help the transport layer to adapt to the injected semantic as well as to payload properties, as interpreted by the application. At the receiver side, the deployment of error handling can thus be distributed between the application and the transport layer in that the application will enable a transport layer activity.

2.6 Conclusions and future work

A reliable multicast protocol most often means that no packet must be lost. This is accomplished either by repeating the information on request, or by sending parity data along with the original information. These methods are most often combined with NAK avoidance schemes. Such schemes can limit the number of messages that are generated in the error handling process by probing either the sender or the receivers, thus enabling the error handling schemes and lessen the burden of the corrective measures. By building trees for control or corrective information, inhomogeneities can be handles to a certain extent. Probabilistic approaches handle this even better.

It would seem that reliability can be a relative expression, which can most favourably be interpreted in the context of the application semantics. This implies that in some contexts, the requirement for reliability is that every single packet be received. In other contexts, this might not be the case, and the absolute reliability becomes a special case of a “semantic reliability”.

In order to be able to use the application specific semantics, the application level framing principle will be explored. As knowledge about the application semantics is present in the application layer, there is also where the processing is most expensive. Error handling is more efficient in the transport layer, where the application semantics unfortunately is not present.

By striving to handle the errors at the transport layer, but configure and tune the protocol from the application, we distribute the computing burden which on one hand relieves the application, and on the other hand caters for a more effective protocol, since each decision will be handled at a lower layer.

It is possible that this technique can be applied also to congestion control.

The most imminent work is now to implement a scheme according to the proposal in a network simulator. Based on the results from the simulations, a live implementation will be made.

Acknowledgements

This work is supported by Ericsson Erisoft AB and the Centre for Distance-spanning Technology.

Bibliography

[1] S. Floyd, V. Jacobson, S. McCanne, C.G. Liu, and L. Zhang, "A reliable multicast framework for light-weight sessions and application level framing," *Computer-Communication-Review*, vol. 25, no. 4, pp. 342–56, Oct. 1995.

[2] Y. Chawathe, S. McCanne, and E. A. Brewer, "RMX: reliable multicast for heterogeneous networks," *Proceedings IEEE INFOCOM 2000*, vol. 2, pp. 795–804, 2000.

[3] Holbrook HW, Singhal SK, and Cheriton DR, "Log-based receiver-reliable multicast for distributed interactive simulation," *Computer-Communication-Review*, vol. 25, no. 4, pp. 328–41, Oct. 1995.

[4] Xiao Z. and Birman K.P., "A randomized error recovery algorithm for reliable multicast," *Proceedings IEEE INFOCOM 2001*, April 2001.

[5] S. Floyd, V. Jacobson, C. G. Liu, S. McCanne, and L. Zhang, "A reliable multicast framework for light-weight sessions and application level framing," *IEEE/ACM-Transactions-on-Networking*, vol. 5, no. 6, pp. 784–803, Dec. 1997.

[6] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," *Computer-Communication-Review*, vol. 28, no. 4, pp. 56–67, 1998.

[7] J Pereira, L Rodrigues, and R Oliveira, "Semantically reliable multicast protocols," *Proceedings of 19th IEEE Symposium on Reliable Distributed Systems*, pp. 60–9, 2000.

[8] R. Rodrigues, R. Baldoni, E. Anceaume, and M. Raynal, "Deadline-constrained causal order," *The 3rd IEEE International Symposium on Object-oriented Real-time distributed Computing*, Mar. 2000.

[9] R. Baldoni, R. Prakash, M. Raynal, and M. Singhal, "Efficient Δ-causal broadcasting," *Journal of Computer System Science and Engineering*, 1998.

Part 3

Applying Semantic Reliability Concepts to Multicast Information Messaging in Wireless Networks

Applying Semantic Reliability Concepts to Multicast Information Messaging in Wireless Networks

Stefan Elf and Peter Parnes
Luleå University of Technology
Centre for Distance-spanning Technology
Department of Computer Science
SE-971 87 Luleå, Sweden
{`self, peppar`}`@cdt.luth.se`

Abstract

Public use of the Internet increases and applications suitable for multicast distribution grow more popular. An increasing percentage of users require wireless connectivity. Multicast solutions must therefore be able to handle receiver and network link heterogeneity. This paper proposes an approach to handle such challenges.

According to a definition of semantic reliability, the reliability concept can be interpreted in terms of application semantics. By using semantic reliability traditional reliability constraints can be relaxed. Our approach is to use an application - transport layer communication to implement a dynamically configurable transport layer protocol whose error-handling rule set can be configured from the application or even from the sender in-session. It can also be initiated from the sender when the session starts.

3.1 Introduction

Protocols for mass-distribution of information have been researched during the last two decades. Although multicast scenarios mostly involve one-to-many situations, such as lecturing or advertising, there is also a growing class of group collaboration applications, which operate on a many-to-many basis.

Networking environments have until recently been more or less static. This situation is now changing as wireless network access is gaining public interest and availability. With the freedom of wireless connectivity comes a number of problems among which the most notable are that multicast trees will need to change more often as the user moves, that link capacity will fluctuate due to the nature of radio propagation, that user equipment will have restricted capacity both in terms of power and memory space.

It seems reasonable to assume that more widely available network connectivity will drive a need for many-to-many communications. Among such collaborative applications are shared network editors, on-line gaming, simulation applications, and shared whiteboards. Many of these applications require reliable services.

In MobileCity, an EU funded ongoing project in Skellefteå in northern Sweden, the International Fair Navigator is being developed. Visitors will have information about products and services, maps etc., stored in a personal digital assistant (PDA). The PDA will also receive information via multicast and users may interact. Since PDA's are connected with a wireless interface and users are highly mobile, they are subject to varying bandwidth and frequent disconnections (Figure 3.1). A reliable service is required to ensure content delivery. According to our approach, it is possible that reliability can be achieved even though all information does not reach the PDA.

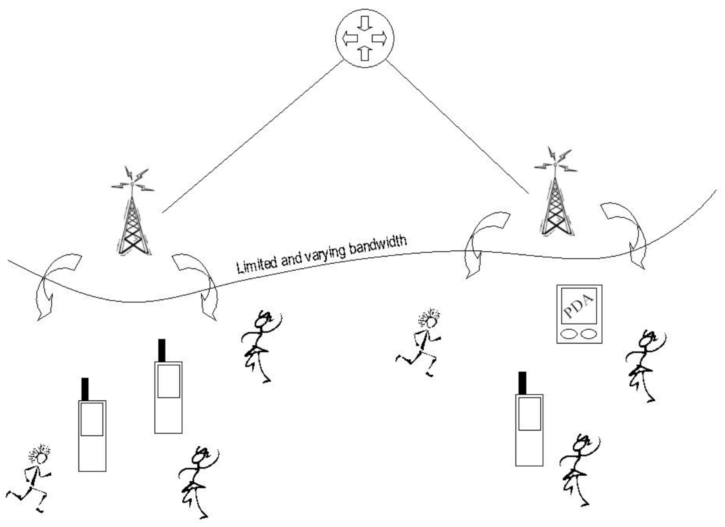

Figure 3.1: In a wireless scenario, excess traffic must be avoided.

A number of reliable multicast protocols have been designed during recent years in which commonly, the transport layer handles reliable services. Each protocol tends to address specific application requirements. Application Level Framing (ALF), first proposed by Clark and Tennenhouse [1], puts the responsibility with the application itself to handle end-to-end reliability. The advantage is that the application can adapt gracefully, while the drawback is that this must be implemented specifically in each application. ALF has grown in popularity shown e.g. by Chawathe et al. in RMX [2], which operates in the transport layer as well as in the application layer.

It is required that a reliable multicast protocol be scalable regardless of whether errors appear or not. It must be efficient for two users or for two thousand users alike. Control message traffic, delay, jitter, and buffer requirements must not cause the protocol to use most of the available bandwidth for control data.

Topological features such as hierarchic levelling, grouping, and clustering have been explored in order to handle the flow of control messages. Other approaches include use of multicasting control messages using multiple multicast groups to layer control message information.

Protocols handle errors either by detecting information loss and resending lost data or by transmitting redundant or parity information with the original data. Some transmit parity information only when a loss is detected. If repair packets are multicast to the group, each packet may repair several lost packets for different receivers.

The ALF approach will allow the application to handle most of these problems. ALF enables error concealment strategies. The drawback is that effectively a new protocol must be developed for each application.

We believe that a welldefined dynamic behaviour in the transport layer, configured by application level information, will enable it to use application semantic properties. We will build on previous work on semantic reliability and attempt to find a straightforward way to formalize the implementation in an application independent way.

The remainder of the paper is structured as follows. Next, we present related work. Current error handling methods are then shortly reviewed. Following, we present a relaxed notion of reliability and discuss a configurable transport layer protocol as a way to implement a semantic reliability. Last, we conclude.

3.2 Related Work

Mauve and Hilt [3], discuss an application-programming interface (API) for reliable multicast supporting distributed interactive multimedia applications. Their approach is close to ours. They propose a subscription-oriented interface in which a sending application controls the use of forward error correction (FEC). The receiving application can specify its (dis-)interest in lost packets. These Quality of Service (QoS) levels can relieve the receiver from having to buffer packets that must otherwise be preserved due to outstanding earlier packets.

The work presented by Pereira et al. [4], Rodrigues et al. [5], and Baldoni et al. [6], is also close to our approach. They explore application level semantic included in the packets.

Pereira et al. [4] propose that as buffer occupancy reaches a high-water mark a constructed semantic in the packet header becomes active. Each packet header contains information pertaining to which earlier packets it renders obsolete. Packets can thus be purged from receiver buffers. Sender or repair server buffers can also be cleaned up using this approach. During congestion receivers may drop all packets that do obsolete packets already in the buffer. During nominal operation this mechanism is inactive. The sender is required to mark all packets that may be discarded, as obsolete.

Rodrigues et al. [5] use message deadlines as a criteria and Baldoni et al. [6] assign a lifetime to each packet. It can be purged when it reaches a deadline or its lifetime has expired.

These obsolescence techniques and the efficiency of them, depend on application semantics. They also rely on the sender to provide the semantic.

Our approach differs from the referenced work in that we act at the transport layer and will create a ”tunnel” between the application and transport layers. This allows the sender application to feed information to configure the receiving side transport layer. The receiving application will be able to configure the transport layer based on its knowledge of incoming payload. The semantics used are associated with application behaviour rather than QoS levels. This will decouple the specific application's subjective involvement from lower layer error handling.

Where nodes are connected wireless to a fixed network or central server, it is also important that configuration actions not only be restricted to the local host, but to the nearest upstream router or server. This will assure that the least amount of traffic will pass the air interface.

3.3 Error Handling

Reliable multicast protocols are either reactive or proactive. Reactive protocols must have a mechanism to detect packet loss. Sender oriented protocols require receivers to acknowledge (ACK) receipt of each packet as in RMTP [7]. With a large receiver group the sender could risk being swamped by ACK's from all receivers resulting in an ACK implosion. The sender must keep state for all receivers and must buffer transmitted packets until all receivers have ACK'ed them.

In a hierarchy the number of control messages reaching the sender is lower. Each ACK needs to travel only to the next higher level in the tree. The host that receives the ACK's issues repair packets for the underlying subtree, which further lowers traffic in the multicast group and allows the sender to share the load with local repair servers.

To handle the scalability problem negative acknowledgements (NAK) are used. As each receiver is responsible for detecting its own lost packets, there will not be any significant load on the sender and the result is improved scalability. In NAK based protocols lost packets are detected using time-outs and message sequence numbers. Lost trailing packets are detected with session messages issued by a receiver announcing the currently largest packet sequence number.

NAK implosion may result from a large number of receivers experiencing simultaneous loss. In Scalable Reliable Multicast [8] a scheme using timers causes a receiver to back off before a NAK is sent, in hope for another receiver to multicast a corresponding NAK. Multicast of control messages and repairs increases efficiency and lowers the chance for an implosion.

Hierarchy, grouping, or clustering of hosts increases efficiency in NAK-based protocols as well as in ACK-based.

In a proactive approach redundant information is transmitted with original data. Ideal proactive protocols require no feedback, which can be true in a real life situation only up to a certain level of packet loss. The cost for this convenience is a constantly higher data rate. A possible need for blocking data will introduce delays in the data transmission.

The Digital fountain approach discussed by Byers et al. [9] is a special application of efficient erasure codes. With these tornado codes a receiver need only receive any subset of packets among those sent, in order to be able to decode original data. If the sender keeps transmitting packets, there is no need for feedback at all. The penalty is again a slightly increased bandwidth demand.

In practice, proactive protocols must be combined with reactive. Hybrid protocols for reliable multicast include RMX [2], where UDP multicast is combined with unicast TCP traffic.

Table 3.1 summarises some challenges, which must be met by multicast protocols in general.

Table 3.1: Multicast protocol challenges

Issue	Remedy	Affected by semantic reliability
Receiver heterogeneity	Grouping, hierarchy, worst case	Yes
Network heterogeneity	Grouping, hierarchy, worst case	Yes
Scalability	Grouping, hierarchy	Possibly
Congestion	TCP-like schemes	Possibly
Control message implosion	Hierarchy	Possibly
Application independence	-	Yes

3.4 Relaxing Reliability

Applying semantics is not sufficient to implement a semantic reliability concept. We need a framework that explores semantics and decouples the application from actual packet handling.

3.4.1 Semantic Reliability

The basis for the idea of semantic reliability [4] is a redefinition of reliability as such. By looking at the data from the application's perspective we can make probable that it is not necessary to receive all information for the application to perform adequately. Thus, we can view a semantically reliable transport to be a transport that is reliable in relation to application semantics.

Configuration information is either distributed in a dedicated packet type or downloaded from the application. It contains a metasemantic describing the type of semantic used, and possible parameters. The specific kind of semantic depends on the application, as shown in Table 3.2.

Each semantic is related to a set of parameters describing the desired behaviour of the transport layer, with respect to each received packet. The most strict semantic is ABSOLUTE, which corresponds to unrelaxed reliability where every packet must eventually be delivered

Table 3.2: Example semantics and properties

Semantic	Meaning	Property	Configured by
ABSOLUTE	All packets received	None	Sender/receiver
OBSOLESCENCE	Packets made obsolete	Seq.no.	Sender
REDUNDANCY	Redundant packet	seq.no.	Sender
LIFETIME	Packet life time	Time	Sender
EXPENDABLE	No crucial information	Boolean	Sender
DEADLINE	Latest arrival time	Time(stamp)	Receiver
CATCH-UP	Disregard packets	Seq.no.	Receiver
BEST-EFFORT	Fully relaxed	None	Receiver

to each receiver. The least strict semantic is BEST-EFFORT - ”do not re-send lost packets”. Although there seems to be a ”range” from ABSOLUTE to BEST-EFFORT, this is not the case. Different semantics do not represent a scale of higher or lower QoS, but rather different aspects of application data properties.

ABSOLUTE and BEST-EFFORT are mutually exclusive. LIFETIME and DEADLINE seem very similar, though the sender sets the former by marking packets, and the latter is determined by the receiving application. OBSOLETE packets are those whose information has been invalidated by more recent packets. EXPENDABLE packets carry information that can be discarded in the first place and need not have any relation to other packets. The CATCH-UP semantic will let a receiver disregard all packets up to a specified packet sequence number, possibly a very large amount of information. The receiver takes this decision. When the receiver has caught up, this semantic is reset.

The packet sequence number, as single numbers, lists of numbers, or ”all numbers less than”, is used to identify obsolete and redundant packets.

When the sender specifies a semantic it is included as information in the packets transmitted to the receiver. The receiving application informs the transport layer of the presence of the semantic. Properties that are conveyed in the packets are presented in Table 3.2.

An application can always override a sender’s recommendation. Not activating a semantic constitutes a more strict behaviour. If a sender specifies ABSOLUTE, a receiver may take a more relaxed position, and specify BEST-EFFORT or in some cases, CATCH-UP, e.g. when a whiteboard page is being updated and the receiver knows that it already has a consistent view.

3.4.2 A Semantic Reliability Framework

There is a trade-off between efficiency of handling errors in the application according to the ALF principle, and efficiency of handling errors at a lower protocol layer, where the necessary information may not be available.

With support from the application layer, the transport layer can handle errors effectively. We propose a model by which the application defines an error-handling rule set based on received data and knowledge of application semantics. This information is then fed to the transport layer. Configurable parts of the transport layer protocol will not be engaged in interactions with other hosts. The transport layer may be differently configured at each host. Therefore, a failing host can refrain from disrupting the session for other participants.

The configuration of the transport layer is done dynamically. The sender supplies the semantics. It can also be constructed by the receiving application, depending on local conditions. When conditions for the receiving application change, the transport layer will immediately be reconfigured.

Receivers who join a session late and wish to update their dynamic transport layer configuration do so by multicasting a request message to its neighbours. The configuration is then multicast to the same scope. Receivers who do not have the semantic parameters can still receive from and transmit to the group. The configuration is not a prerequisite, but rather an enhancement, a protocol booster, which can be used with any protocol.

If we extend this framework to let the application configure the nearest upstream router's transport layer it will be even more versatile. With wireless connectivity this is more important, since we must not send unnecessary data over a wireless link.

Semantics distributed by the application will be interpreted directly by the transport layer, which is then configured accordingly. The application will configure the transport layer using an API. This API can also be used by the application to query for sender-initiated semantics. The implementation is divided into two parts. The application level uses the first part to set up the semantics.

```
Semantic wbSemantic = new Semantic("channelId",
         SemanticProperty.BEST_EFFORT);
```

The transport level executes the semantic rules, using the other part

```
theSemantic = theSemantic.getInstance("channelId");
if (theSemantic.isBestEffort) {
  // Do not handle lost packets
  return;
}
```

The channelId associates the data flow with the relevant semantics.

Some of the semantics will work properly only when the sender adapts the packet payload size to payload characteristics. In a video application, each packet should hold one or more full frames, depending on the frame type. In a stock exchange update application, each packet should convey one record of information.

Fair Navigator

The Fair Navigator scenario uses a centralized information service. People carry Fair Navigators (PDA's) with suitable software. Since these have limited storage and computing capacity, and are connected through a wireless interface, it is important that there is no excess traffic.

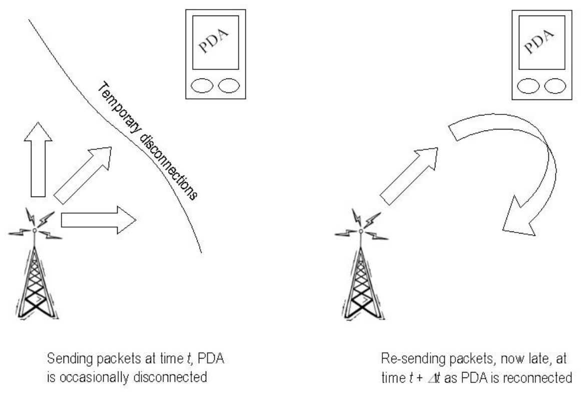
PDA
Temporary disconnections
Sending packets at time *t*, PDA is occasionally disconnected
PDA
Re-sending packets, now late, at time $t + \Delta t$ as PDA is reconnected

Figure 3.2: PDA transport layer drops outdated packets.

When the information regarding a meeting is transmitted, the sender will apply the LIFETIME semantic. When the meeting is over, it makes no sense to accept the packets (Figure 3.2). A PDA coming on-line after a break detects expired packets and can use the CATCH-UP semantic to discard these.

If the connection degenerates, the application can use the BEST-EFFORT semantic in order to minimize the network impact and to make the application perform as good as possible.

3.5 Shared Whiteboard

In a shared whiteboard packets must not be lost. A user draws a circle, followed by other figures. Another user has lost the circle-packet. Before this error is corrected, the first user deletes the circle. Now, there is no meaning in re-sending the original circle-packet.

The sender can detect this and remove the original circle-packet from its buffer. The receiver, still miss-ing both packets, can upon reception of the delete-packet remove the request for the originally lost packet and drop the delete-packet since it will point the circle packet out as being obsolete.

A complete update of a page can be detected by the receiver and if the page is already up to date, the receiver can CATCH-UP and drop all update packets.

3.6 Conclusions and Future Work

In wireless connections it is important not to send data that the application can manage without. If absolute reliability is required large delays can be introduced because of the greater chance for disconnections and varying bandwidth. If a protocol must adapt to a worst-case receiver without semantic reliability in a wireless network, the better nodes will suffer badly.

Current solutions explore semantic reliability but do not devise a complete solution to the problem of reducing the number of messages, which are due to the error-handling process itself. The overall number of messages can be reduced using hierarchic models, but in the end, the wireless receiver failing to receive packets will cause problems where it hurts the most, in the wireless link.

In order to be able to use application specific semantics, we explore the application level framing principle in the sense that the application is involved in the process. The transport layer will handle packets more effectively, but the application will decide which packets can be dropped with the least concern. Our approach will enable applications to adapt to varying resources such as in the Fair Navigator example, where a number of wireless connected, ”network friendly”, PDA’s located at weak spots will act in both in the best interest of the user and of the wireless network.

We believe that this technique can be applied also to congestion control. It will give the application some control over which packets are dropped from a queue that is full or almost full, based on application semantics.

The first implementations of this framework are planned in the Fair Navigator and is under way in the shared whiteboard of the distributed collaboration software MarratechPro [10].

Acknowledgements

This work is supported by Ericsson Erisoft AB, Centre for Distance-spanning Technology, the Swedish Research Institute for Information Technology, and Mäkitalo Research Centre.

Stefan Elf is also with Ericsson Erisoft AB, SE-931 87 Skellefteå, Sweden. Views expressed in this paper are his own and not necessarily shared by his employer.

Bibliography

[1] D. Clark and D. Tennenhouse, “Architectural Considerations for a New Generation of Protocols,” in *Proceedings of SigComm*, Philadelphia,PA, 1990, pp. 201–208.

[2] Y. Chawathe, S. McCanne, and E. A. Brewer, “RMX: reliable multicast for heterogeneous networks,” *Proceedings IEEE INFOCOM 2000*, vol. 2, pp. 795–804, 2000.

[3] M Mauve and V Hilt, “An application developer’s perspective on reliable multicast for distributed interactive media,” *Computer-Communication-Review. vol.30, no.3; July 2000; p.28-38*, 2000.

[4] J Pereira, L Rodrigues, and R Oliveira, “Semantically reliable multicast protocols,” *Proceedings of 19th IEEE Symposium on Reliable Distributed Systems*, pp. 60–9, 2000.

[5] R. Rodrigues, R. Baldoni, E. Anceaume, and M. Raynal, “Deadline-constrained causal order,” *The 3rd IEEE International Symposium on Object-oriented Real-time distributed Computing*, Mar. 2000.

[6] R. Baldoni, R. Prakash, M. Raynal, and M. Singhal, “Efficient Δ-causal broadcasting,” *Journal of Computer System Science and Engineering*, 1998.

[7] JC Lin and S Paul, “RMTP: a reliable multicast transport protocol,” *Proceedings of IEEE INFOCOM ’96*, vol. 3, pp. 1414–24, 1996.

[8] S. Floyd, V. Jacobson, S. McCanne, C.G. Liu, and L. Zhang, “A reliable multicast framework for light-weight sessions and application level framing,” *Computer-Communication-Review*, vol. 25, no. 4, pp. 342–56, Oct. 1995.

[9] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, “A digital fountain approach to reliable distribution of bulk data,” *Computer-Communication-Review*, vol. 28, no. 4, pp. 56–67, 1998.

[10] Marratech, “- The e-meeting company,” URL[1], October 2001, Visited September 9th, 2002.

[1] <http://www.marratech.com/>

Part 4

Efficient Workspaces through Semantic Reliability

Efficient Workspaces through Semantic Reliability

Jeremiah Scholl and Stefan Elf and Peter Parnes[1]
Luleå University of Technology
Centre for Distance-spanning Technology
Department of Computer Science
SE-971 87 Luleå, Sweden
{jeremiah, self, peppar}@cdt.luth.se

Abstract

Transport level protocols typically provide "best-effort" data delivery where no attempts are made to recover lost packets, or "reliable" data delivery where techniques are used to ensure that all data sent out eventually reaches the receiver(s). However, it has been suggested that some applications will perform better when using dynamically configurable reliability based on application level semantics. A general framework has been created with the intent of delivering this type of semantic reliability to a wide variety of applications, but to date little research has been done to demonstrate how any applications can benefit from the use of such a framework. This paper addresses this problem in that it discusses semantic reliability in the context of collaborative workspaces.

4.1 Introduction

IP-Multicast provides efficient one to many best-effort delivery. However, a wide variety of reliable multicast protocols have been introduced over the last several years because some applications perform poorly when they suffer from packet loss. While designed around separate application requirements and network constraints these protocols have in common that they are engineered to guarantee that each packet transported eventually reaches its designated receiver. A comprehensive study of multicast transport protocols including the reliable transport protocols mentioned above is given by Obraczka [1].

Alternatively, it has been suggested that some applications could benefit from more relaxed reliability rules where only a subset of packets lost during transmission would be recovered. Ideally this would save bandwidth by reducing the amount of repairs sent on the network that recover data, which is no longer useful to the receiver. This idea has not been explored as extensively as "absolute" reliability but several ideas have been presented for solving specific application scenarios by using the technique. In addition, a general framework has recently been described which aims to make relaxed reliability based on application semantics available to a wide variety of applications (see section 4.2).

Little work has been done to date in order to demonstrate how this type of framework can enhance specific applications. This is in practice a potentially time consuming process be-

[1]0-7803-7661-7/03/$17.00©2003 IEEE

cause the semantic rules that determine when to send repairs must be worked out separately for each application. Despite this drawback, the potential benefits in terms of efficient use of bandwidth and stability when facing packet losses are compelling. In particular this technique could be very useful when deploying distributed multimedia applications on wireless telecommunication networks, as these links tend to be lossy and have limited bandwidth.

This paper addresses this issue by examining semantic reliability in the context of IP-Multicast collaborative workspaces, which are complex applications for enabling group communication. Focus on a specific implementation is required before the necessary Application Level Framing (ALF) [2] can be performed due to the fact that collaborative workspaces are inherently different from one another. In order to obtain this focus while maintaining a real-world view, an existing partnership between Luleå University of Technology and Marratech AB has been utilized. Thus, the analysis in this paper focuses on features contained in Marratech's commercially available collaborative workspace implementation, Marratech Pro [3]. This will help this paper serve developers of similar applications who are interested in an analysis based on a highly deployable solution because Marratech Pro is based on the well known and the widely used protocols RTP [4] and SRM [5] as well the standardized video codec H.261 [6].

4.2 Related Work

Several approaches exist for using application or data semantics in order to classify or handle information. Mauve and Hilt [7] have presented an Application Programming Interface (API) for reliable multicast that allows the sender to control the use of forward error correction (FEC) in a subscription-oriented interface, which allows a receiver to proclaim its interest, or lack thereof, in lost packets. Several other authors have also explored having the sender include application level semantic information in packets. For example, Pereira et al.. [8] describe a method where a buffer occupancy monitor is combined with an obsoleteness marker in packet headers in order to purge redundant messages from buffers or to immediately drop incoming packets. This allows a buffer to make space for useful, non-obsolete packets but has also the drawback that the sender must mark all packets accordingly.

Approaches based on buffering and FEC can however be viewed as problematic when implementing them in collaborative workspaces. This is because buffering introduces delay, which can harm interactivity between users, and the abundant use of FEC introduces overhead into the data stream, which harms scalability by reducing the number of effective senders that can participate in a session. The solutions described in this paper thus focus on using application semantics along with recovery-based reliability, as this is low-weight compared to FEC without introducing the interactivity problems associated with buffering. Other approaches for semantic reliability include different packet header markers for scenarios other than obsoleteness. These include message deadline as in Rodrigues et al. [9] and the similar lifetime semantic in Baldoni et al. [10].

Elf and Parnes [11] have presented a robust framework based on several semantics, which suits our needs for classifying and deploying semantic information to recovery based feedback. The approach described focuses specifically on how to classify data based on different application scenarios, which are ”tunneled” down to the underlying reliable multicast protocol in use by the application. This enables the framework to be used with a wide variety of protocols and applications.

The framework is also flexible in that the sender can distribute configuration information in data packets (the semantic would alter the behavior of all receivers) and each receiver can independently configure their own behavior based on specific local application properties or network conditions. Therefore, the semantics used are associated with application behavior rather than QoS levels. Because each host can be configured independently of all other hosts it offers the advantage that a complex application with several media (for example a collaborative workspace) can be dynamically configured to perform in an optimal way.

The framework describes several semantics in which the traditional view of reliable delivery is considered a special semantic case, referred to as ”absolute”. Similarly the opposing view in which no attempts are made to retransmit lost packets is labeled as ”best-effort”. In addition, several other semantics are available, which can be used to benefit an application in situations where neither the ”absolute” nor ”best-effort” semantic seems to provide optimal performance. Table 4.1 provides a list of possible semantics, which may be useful in different scenarios. The semantics listed in the table are those that were identified in the reference publication. However, the framework was intended to be expandable so other semantics may be utilized when necessary.

Table 4.1: Example semantics and properties (from[11]).

Semantic	Meaning	Property	Configured by
ABSOLUTE	All packets received	None	Sender/receiver
OBSOLESCENCE	Packets made obsolete	Seq.no.	Sender
REDUNDANCY	Redundant packet	seq.no.	Sender
LIFETIME	Packet life time	Time	Sender
EXPENDABLE	No crucial information	Boolean	Sender
DEADLINE	Latest arrival time	Time(stamp)	Receiver
CATCH-UP	Disregard packets	Seq.no.	Receiver
BEST-EFFORT	Fully relaxed	None	Receiver

Each semantic is intended to relate to one of several parameters that together describe the desired behavior of the transport protocols, with respect to each packet. Certain semantics may exclude others, such as ”absolute” obviously is mutually exclusive to ”best-effort”. Some semantics relate to the sender, such as ”lifetime”, while others relate to the receiver, the example being ”deadline”.

The next section discusses Marratech Pro in the context of this framework. Several ideas are presented, which attempt to save bandwidth, reduce latency or provide stability through the use of semantics other than the traditional ”absolute” and ”best-effort”. This is intended both to provide a view of how more efficient collaborative workspaces can be implemented in the future and help the reader obtain a clearer picture of how relaxed reliability can be utilized to create more efficient applications in general.

4.3 Semantic Reliability in Collaborative Workspace Tools

Marratech Pro is a commercially available collaborative workspace, which offers common and intuitive tools for enabling group communication. These tools are interactive audio, video, chat, whiteboard, and a shared web-browser. Currently, with only one exception, Marratech Pro classifies data in the traditional manner treating audio and video as ”best-effort” while the browser, chat, and whiteboard are viewed as needing ”absolute” reliability, which they obtain from the use of an SRM-like [5] mechanism.

The one example of relaxed reliability based on application semantics that has already been implemented into the application occurs when one or more receivers require information about whiteboard pages after reestablishing contact or joining a session for the first time. In some situations these updates end up being sent to the entire group because collaborative workspaces need to run in environments with Network Address Translators, firewalls and reflectors, which will result in all traffic being sent via multicast [12]. Thus, this will cause some participants in the session to inevitably receive packets describing whiteboard pages for which they have already obtained an up-to-date copy.

When this happens, hosts that are not interested in the update ignore losses and do not generate NAKs for missing packets related to the update. It is important to note that this feature has existed in Marratech Pro for several years, which illustrates that the idea of using relaxed reliability is not new by any means. However, it is equally important to note that this behavior can be obtained by utilizing the ”redundancy” semantic associated with the framework. This demonstrates that the framework is flexible enough to incorporate old as well as new ideas, which can be valuable when adding it to existing applications.

The remainder of this section examines additional ideas for how to provide optimized reliability based on application semantics. More specifically, several features of the whiteboard, shared web-browser and interactive video streams provided by Marratech Pro are discussed and concrete examples of how the framework can be applied to enhance these tools are given.

4.3.1 Minimizing transmission of Obsolete Data from Shared Web-Browsers and Whiteboards

One common tool provided by collaborative workspaces is a shared web-browser, which is used in order to provide an effective method of distributing presentation slides. This is accomplished by allowing a user to control the shared browser so that each page it loads is automatically multicast out to the group. The presenter can then load a slide into each user’s browser by simply clicking on a hyperlink or typing a URL.

At times congestion or other factors may result in the presenter moving on to a new slide before all the participants have finished receiving the current page. For example, this will occur if the presenter follows several hyperlinks in order to reach the page that she intends to discuss with the group. In this situation continuing to repair packets for the old pages may be unnecessary as this data can be viewed as obsolete from a presentation standpoint. By applying the ”obsolescence” semantic from the framework these useless repairs can be avoided, which will free up bandwidth and allow the current slide to more quickly reach the group. Table 4.2 describes two semantic rules that apply this idea.

Both of these rules can be used at different times in order to benefit the application in different scenarios. For example, use of the first rule will limit all data recovery to packets for the current slide only. This will keep bad receivers from clogging the network with increasing amounts of control data as each new slide is loaded.

On the other hand, during some deployments the users may prefer to allow each receiver to act independently and would then employ the second rule only. This would allow a particular host to free up bandwidth for the current slide without affecting other hosts. This situation would also allow a host to have the option of replaying the presentation later on by attempting to recover the previous slide if for example a reference to this slide was made by the user clicking the back button in her browser.

Table 4.2: Semantics for reducing the retransmission of obsolete data in a shared web-browser.

Semantic	Use
OBSOLESCENCE	The sender will stop repairing packets for old pages as soon as it begins to send a new page.
CATCH-UP	When a new page begins to arrive a receiver will stop sending nack's for packets representing old pages.

A similar optimization can be made to whiteboards, which are included in collaborative workspaces in order to give the participants the ability to create and dynamically manipulate shared slides. Whiteboards are generally object-based, allowing the users to create and delete pages that can contain many objects. The objects themselves can be created, modified, or deleted at any time, and can consist of several different forms which may include hand drawn curves, simple geometric shapes, text, and images.

Because pages and objects can be deleted at any time it is possible for recovery packets to be sent out, which describe modifications for an object or page that no longer exists. For example, this can occur if a page is deleted while packets describing modifications to objects on the page are still en route to members of the session. A host could then avoid resending data referring to objects or pages that it identifies as deleted by using the ”obsolescence” semantic in a similar way to that described for shared web-browsers.

However, unlike with a shared browser it may not be obvious to every host that the information being requested is in fact obsolete. Therefore, a host should send out an ”obsolescence” message to the group when it would normally send a repair. This will keep hosts from continuing to send NAKs when their SRM back off timer has expired. In addition, this will be more efficient than sending a repair because these messages can be quite small compared to data packets and one message will be able to declare a large number of packets as obsolete.

4.3.2 Whiteboard text-node semantic reliability using local dictionary

Another idea for applying semantic reliability to a white-board application occurs when text is written into a page. Often whiteboards (including Marratech Pro) implement this feature by multicasting each letter individually to the users while the user is typing the text. While this does introduce a fair amount of overhead in terms of packet headers it allows the participants in the session to see each word as it is being added to the whiteboard. Because letters are sent individually an interesting possibility for packet recovery based on application semantics becomes available, which would not be available if each sentence were sent individually as is usually the case with chat tools. This becomes apparent because at times it may be possible to accurately guess a word that has been only partially received by using a local dictionary as well as the semantics of the adjacent words. In this situation it will not be necessary for missing letters to be repaired because enough information has already been received to reconstruct the complete word at the receiver. By avoiding repairs, in addition to saving bandwidth, another benefit is that the amount of time it takes for a receiver to gain a view of the word transmitted will be decreased. This could be especially useful when using a whiteboard on a wireless link where a fair amount of uncorrelated losses are often experienced.

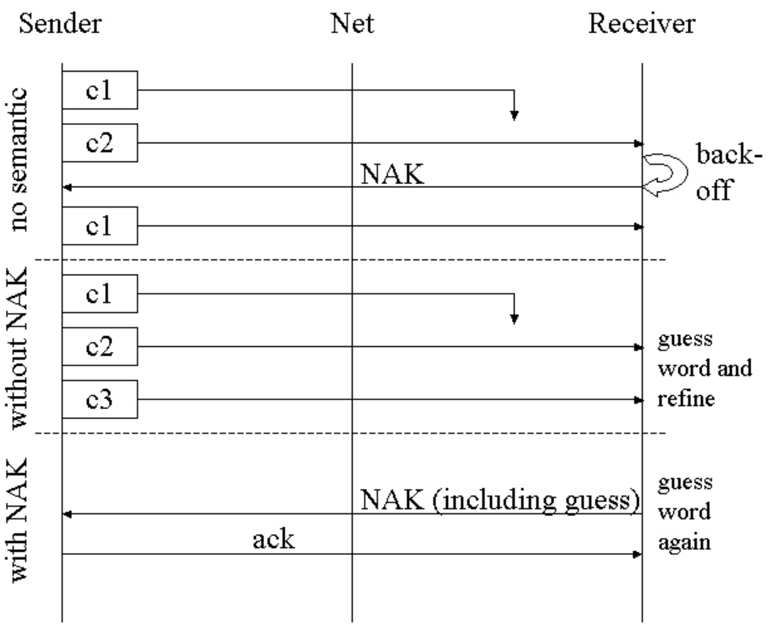

Figure 4.1: The principle for text-node semantics using a local dictionary.

Of course a word that has been guessed by a local dictionary may or may not be correct. This fact must be appropriately displayed for the user for example by changing some display attribute. In addition, the local host must not use a guessed word when sending repairs to other hosts until the exact word has been confirmed.

Two separate scenarios for avoiding the retransmission of lost letters are illustrated in Figure 4.1.

In the *without NAK* scenario the receiving host does not send NAKs for missing letters after a guess. This will be quite effective when the word guess is correct. However, even if the dictionary cannot give a correct guess it will still be effective if enough information was made available from the received letters for the human user to deduce the correct the word, in which case a NAK would still be unnecessary.

In the *with NAK* scenario, a NAK is sent in order to verify that the dictionary guess is correct. The current guess for the word is included in the NAK in order to accomplish this. Then, after receiving a NAK with a guess a host can then simply respond whether the guess is correct or not. When the guess is correct then the receiver can choose not to NAK any additional losses for the word as it already has enough information to display the correct word to the user. When the guess is not correct, the entire word can be re-sent in one packet to the receiver.

4.3.3 Efficient Reduction of Error-Propagation in Multicast Interactive Video

While the ideas presented above deal with the more natural idea of using application semantics in order to make traditionally ”reliable” media perform more efficiently it is also interesting to discuss semantics in the context of media that are normally treated as ”best-effort” by developers. Rather than making these media run more efficiently, semantics instead open up the possibility to introduce error control techniques that are normally thought of as grossly inefficient. Such is the case with real-time video streams, which are commonly provided by collaborative workspaces in order to give the participants a visual presence of each other and their surroundings.

One of the most challenging aspects in obtaining high quality interactive video is that many commonly used error control techniques for non-interactive video are unavailable because they introduce latency in frame playout times. For example, without the benefit of buffers, repairs typically arrive too late to be displayed in the intended frame. An alternative is to use FEC, but as stated in section 4.2 this can consume a considerable amount of bandwidth, reducing the number of effective senders that can participate in a multi-sender session.

However, repairs can be used with interactive video in order to reduce error propagation even if they do not arrive in time to be displayed with the current frame. This technique has been implemented for unicast MPEG traffic and has shown to be low-weight and effective [13]. With a bit of additional design work this concept can be used in order to reduce error propagation for multicast H.261 traffic as modeled in Figure 4.1.

However, unlike with unicast traffic, NAK suppression will need to be utilized in order to keep the necessary loss-feedback for repairs from impacting scalability. This introduces a problem because today’s most deployable NAK-suppression technique is based on random

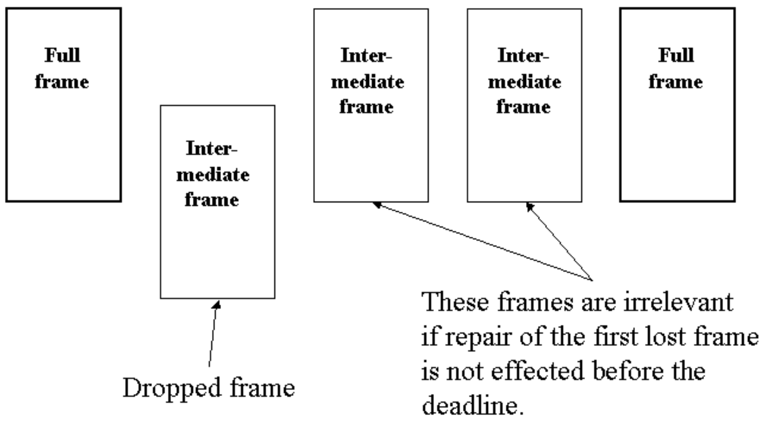
Full frame
Inter-mediate frame
Inter-mediate frame
Inter-mediate frame
Full frame
Dropped frame
These frames are irrelevant if repair of the first lost frame is not effected before the deadline.

Figure 4.2: The principle for text-node semantics using a local dictionary.

timers [5], which introduce delay in reporting losses back to the sender. The delay could result in repairs and/or NAKs sent out, corresponding to blocks that have since been updated due to new movement or a periodic conditional replenishment sweep. However, the number of unnecessary repairs can be kept to a minimum if instead of resending every lost packet, additional semantic rules are used as discussed in Table 4.3.

Table 4.3: Semantics for enabling efficient error recovery in interactive H.261 video.

Semantic	Use
OBSOLESCENCE	Before sending a repair the sender will check to see if the block described in the repair has been updated after the original packet was sent. If so, the repair is ”obsolete” and should not be sent.
DEADLINE	In some situations a receiver can detect which block has been lost. A receiver will then avoid sending this NAK if it receives a new packet describing the block before its random timer expires.

These semantic rules will reduce the number of repairs sent which contain obsolete data to a rather small number, opening up an error-control technique that would otherwise be too inefficient to be generally useful.

4.4 Evaluation

Table 4.4 represents the bandwidth savings in bytes, not including control data, and improved latency in seconds when several of the tasks from section 4.3 are performed. Column 1: The task being performed; 2: The number of bytes saved not including control data; 3: The improved latency in reception by the user when considering the bandwidth savings from column 3.

Table 4.4: Savings in bandwidth (bytes) and delay (seconds) experienced by a receiver with a 256 Kb/s link experiencing 10% loss. See text.

Task	Bandwidth	Delay
Slide distribution	200000	6.5
Dictionary (without NAK)	2800	70
Dictionary (with NAK)	1700	40
H.261 repair	-1040	10-15

Slide Distribution. A presenter follows several links in order to load the desired presentation page. Bandwidth savings S, is defined by $S = nbr$, where n is the number of intermediate pages, b is the average bandwidth per page, and r is the loss rate incurred by the receiver. The decrease in latency in receiving the presentation page will be equal to S/D where D is the speed of the client link.

The numbers in the table reflect the savings incurred when 4 intermediate slides with an average size of 500 KB are loaded.

Dictionary. Lexical and semantic properties of words are used to guess the missing letters from a word. Two scenarios are examined, these being the *without NAK* and *with NAK* (see section 4.3.2). Assuming no feedback redundancies the first scenario will result in one packet per loss being saved on the incoming link. The second scenario adds the cost of the NAK carrying the guessed word, and one reply, possibly containing the correct word. This process will save bandwidth over traditional repairing methods because one repair packet can repair several lost letters.

The delay required for the user to view the completed word is calculated based on the assumption that it takes one half round trip time for a packet to reach the receiver. For the without NAK scenario the gain in delay is given by:

$t = t_1 + (1 + p)(t_1 + t_{bk}) + (1 + p)^2(t_1 + t_1) - t_1$

where t_1 is half the round trip time, p is the probability for a loss occurrence, and t_{bk} represents the SRM back-off time.

The figures in Table 4.4 are based on the savings when 100 words of typically 5 characters are written into the white-board as text, with the typical values $p = 0.10$, $t_1 = 0.3$, $t_{bk} = 0.35$.

The with NAK scenario adds very little traffic since complete words are transported rather than characters. The table shows figures based on the assumption that 50% of the guesses are correct. There will be an additional delay for transporting the nack to the sender and for receiving the sender's NAK/ACK regarding the guessed word.

H.261 Repair. A packet containing H.261 data is retransmitted in order to correct future frames. Packet size for H.261 data is dependant on the number of blocks replenished and will vary, as several blocks may be included in one packet. The bandwidth ”savings” in the table are negative and represent the additional bandwidth consumed for the worst-case scenario of a repair transmit equal to the maximum transmission unit for Marratech Pro.

Additionally, improvements in latency will depend on the content of the video stream because movement reduces the amount of time that an effective repair can be delivered. Therefore, the latency value represents a scenario with no moment occurring in affected areas before the next background sweep. The time between sweeps varies greatly depending on many factors so the values shown are representative of typical values observed while running sessions at Luleå University of Technology.

4.5 Conclusions and future work

In this paper we have described how semantic reliability can be applied to a real-world collaborative workspace application. We show, using appropriate examples, that dynamic behavior based on semantics can be useful in various aspects of the workspace. In interactive video, which is normally not handled with reliability, we introduce reliability based on application semantics in order to strengthen the transport to a degree at which the application can benefit from it. For the whiteboard and shared web-browser tools ideas have been presented for relaxing reliability based on application semantics, which will allow the applications to handle errors in a manner tolerable by the user. It is important to note that the default behavior of the framework is always the standard behavior for the application or the protocol in question.

Future work includes implementation of these and possibility other modifications into Marratech Pro and since there is a user base, we intend to perform real-life testing and measuring. This is intended in order to give more concrete conclusions about how the ideas presented will enhance the user experience. While this paper has used more of a ”big picture” view, in the future each idea will be explored into more minute detail. For example, improvements in video quality, which is the main goal of adding recovery based reliability, cannot be measured without testing the implementation on a user base.

Acknowledgements

This work is supported by Ericsson AB, Centre for Distance-spanning Technology, the Swedish Research Institute for Information Technology, and Mäkitalo Research Centre.

Stefan Elf is also with Ericsson AB, SE-931 87 Skellefteå, Sweden. Views expressed in this paper are his own and not necessarily shared by his employer.

Bibliography

[1] Obraczka Katia, “Multicast transport protocols: a survey and taxonomy,” *IEEE Communications Magazine*, vol. 36, no. 1, pp. 94–102, Jan. 1998.

[2] D. Clark and D. Tennenhouse, “Architectural Considerations for a New Generation of Protocols,” in *Proceedings of SigComm*, Philadelphia,PA, 1990, pp. 201–208.

[3] Marratech, “- The e-meeting company,” URL[2], October 2001, Visited September 9th, 2002.

[4] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: A transport protocol for real-time applications,” 1996, IETF RFC1889.

[5] S. Floyd, V. Jacobson, S. McCanne, C. Liu, and L. Zhang, “A reliable multicast framework for light-weight sessions and application framing,” in *ACM SIGCOMM*, 1995.

[6] *Video Codec for audiovisual services at p*64 kbps”, Recommendation H.261*, 1993.

[7] M Mauve and V Hilt, “An application developer’s perspective on reliable multicast for distributed interactive media,” *Computer-Communication-Review. vol.30, no.3; July 2000; p.28-38*, 2000.

[8] J Pereira, L Rodrigues, and R Oliveira, “Semantically reliable multicast protocols,” *Proceedings of 19th IEEE Symposium on Reliable Distributed Systems*, pp. 60–9, 2000.

[9] R. Rodrigues, R. Baldoni, E. Anceaume, and M. Raynal, “Deadline-constrained causal order,” *The 3rd IEEE International Symposium on Object-oriented Real-time distributed Computing*, Mar. 2000.

[10] R. Baldoni, R. Prakash, M. Raynal, and M. Singhal, “Efficient Δ-causal broadcasting,” *Journal of Computer System Science and Engineering*, 1998.

[11] S. Elf and P. Parnes, “Applying semantic reliability concepts to multicast information messaging in wireless networks,” in *IRMA Conference Proceedings: Issues & Trends of Information Technology Management in Contemporary Organizations*. Information Resources Management Association, May 2002, Idea Publishing Group.

[2] <http://www.marratech.com/>

[12] Scholl J and Parnes P, “Low-weight congestion control for multi-sender applications,” *IFIP/IEEE International Conference on Management of Multimedia Networks and Services 2002,* Oct. 2002.

[13] Rhee I, “Error control techniques for interactive low-bit rate video transmission over the internet,” *Computer-Communication-Review. vol.28, no.4; Oct. 1998; p.290-301,* 1998.

Part 5

Applying User-behavior to Bandwidth Adaptations in Collaborative Workspace Applications and Video Conferencing

Note: An minor error has been corrected in this paper compared to the version that was submitted for review. It concerns the pseudo code which describes the bandwidth allocation algorithm in which an equals sign has been replaced with a left arrow.

Applying User-behavior to Bandwidth Adaptations in Collaborative Workspace Applications and Video Conferencing

Stefan Elf and Jermiah Scholl and Peter Parnes
Department of Computer Science & Electrical Engineering, Media Technology Division
Luleå University of Technology
SE-971 87 Luleå, Sweden
{stefan.elf,jeremiah.scholl,peter.parnes}@cdt.luth.se

Abstract

A bandwidth-sharing scheme for group video conferencing is presented in this paper. The key features of the scheme are the monitoring of user behavior and message passing, which are used by each client in order to identify and report their interest in other group members. Each video sender operates on the information about other users' interest in order to adjust the sender's own frame rate, resolution, and ultimately bandwidth consumption in an attempt to satisfy the current interests of the receivers as well as the overall bandwidth constraints of the session. A general framework, an initial prototype, and a bandwidth allocation algorithm are presented together with experimental results. The experiences from the prototype have prompted refinements to the bandwidth-allocation algorithm that will be important for future implementations. We also conclude that the necessary messaging will not add a significant amount of bandwidth.

5.1 Introduction

Throughout history, people have always wanted to interact, to meet and discuss. Already since the advent of the telephone, we have been seeking to collaborate without regard to the sometimes-great distances between us, being able to form groups based not on geographic location, but rather on interests and skills. As electronic networks have emerged and grown, and bandwidth has increased applications that support voice over IP in various flavors are on the verge of replacing the land-line telephone.

Body language is an integral part of how humans communicate. This is why voice-only communication cannot entirely replace real-life meetings. One class of applications, the *collaborative workspaces*, seeks to fill that gap. By offering a number of services such as voice, text-based chat, shared whiteboards, shared web browsing, and video communication, their objective is to create a virtual presence. Even if collaborative workspaces come in various shapes and supplies a number of useful services, their key function is to satisfy the human need for not only hearing the words a person speaks, which is perhaps not even half of the message (according to some authors [13] as little as 30%), but also seeing this person. This conveys the message to a fuller extent. It also, which is possibly as important, creates a *feeling of presence*.

In the same way that the speaker's body language is important for the listener and viewer, it has intrinsic properties relating to the importance of the message that is being conveyed. Since the video streams from the members of a collaborative workspace session are also a major component with regards to bandwidth consumption, bandwidth allocation schemes, and especially dynamic schemes, may become a key feature regarding the cost-effective use of bandwidth.

IP multicast offers scalable media distribution and is an enabler for collaborative workspaces. Although IP multicast has been around since more than 20 years, Internet-wide deployment is not a reality today. On the other hand these kinds of applications are making their way into educational and industry environments as multicast is being deployed in subnets such as in university networks or in corporate networks.

In addition to the variety of implementations of collaborative workspaces, there are also a number of usage scenarios, each of which may warrant their own optimizations. There is for example the lecture scenario in which one person is the sole sender and all other are listeners. In a discussion scenario, any number of session members can be an active sender while another group are again just listeners. The senders shift rapidly.

Yet another scenario is named the *electronic corridor* and is used at Luleå University of Technology on a daily basis. People who may or may not be geographically co-located, form a virtual work-office corridor by being members of a dedicated collaborative workspace session. In this scenario we would at times recognize the lecture scenario, at other times the discussion scenario, and at yet other times, we would find an idle virtual meeting-place where people during periods of time attend to their own business, while just keeping up a visual presence in the corridor.

In the first scenario it would be simple to allocate the major part of the bandwidth to the lecturer, while in the second scenario the most obvious solution would be to share the bandwidth equally among the session members. But since neither scenario, like in real life meetings, is static, such a solution designed for a static case, would fail. This is why floor control is a poor choice of strategy. In most cases there will be not exactly one sender, but at least one sender. In the electronic corridor, we wish to consume a minimum of bandwidth, while being adequately updated on the activities of the corridor inhabitants. Thus, we can appreciate that the number of important video streams will inevitably vary widely over time.

The remainder of this paper is organized as follows. Next, in section 5.1.1 research issues are presented. Section 5.1.2 discusses related work. Then, in section 5.2 we discuss the design of the video bandwidth adaptation scheme, including the identification of important video streams, the proposed algorithm, and experiences from running a prototype in a small group of research people. In section 5.3 we summarize, conclude, and suggest directions for future work.

5.1.1 Research Issues

The key research issues in this work are

- How should applications be designed to be able to cost-effectively distribute media under various usage scenarios and also in heterogeneous environments?

- How can a scheme for dynamic video bandwidth allocation be designed to help applications use video presence while conserving bandwidth?

- Would an adaptive video bandwidth implementation be able to handle the widely varying conditions that we find in different usage scenarios?

The first question is a more general one, also covering error handling and congestion control. The issues discussed in this paper seek to answer a part of that only, and more precisely that of the video distribution. This paper proposes one such scheme for dynamic allocation, and it lays the foundation for answering the third question.

The paper presents a bandwidth-sharing scheme for group video conferencing aimed at a general use collaborative workspace environment, such as the electronic corridor. The scheme operates by first identifying session participants that are of high importance to other group members and then allocating them a larger share of the session bandwidth. This is achieved primarily through the implicit detection of user-behavior, such as the configuration of a user's desktop, and utilizes message passing so that receivers can reflect their interests back to the senders in question.

5.1.2 Related Work

The use of implicit user-behavior in resource control has been applied to a wide range of multimedia applications, with each scheme being limited in scope to a specific domain. For example, Kulju et al. [6] investigated user behavior in the context of video streaming, while Ott et al. [10] focused on its use within their own 3D landscape. In addition, recent work in collaborative workspaces has investigated how hints may be used in order to dynamically control the use of reliable multicast [11]. The work most similar to that presented in this paper is the SCUBA protocol [1], which also uses the detection of user interest in order to allocate bandwidth in video conferencing. SCUBA described the basic architectural components for schemes of this type, but little research has been done in this area since its introduction and new ideas as well as refinement on several points are still possible.

Chen [2] designed a multi-party video conferencing system in which low-frame-rate video was sent during idle periods and the frame rate was accelerated as soon as a user made a gesture, thus signaling relevant activity. In this application the low-frame-rate video was sent reliable on top of UDP. In Chen's system, this reliability was necessary for not overlooking important gestures that in turn was a significant part of the bandwidth control. We propose to use this or a similar scheme as a quality enhancement tool during low-frame-rate in future prototypes.

Though user behavior is the principal component in the work that is presented in this paper, application semantics is an important component for this to work smoothly with the

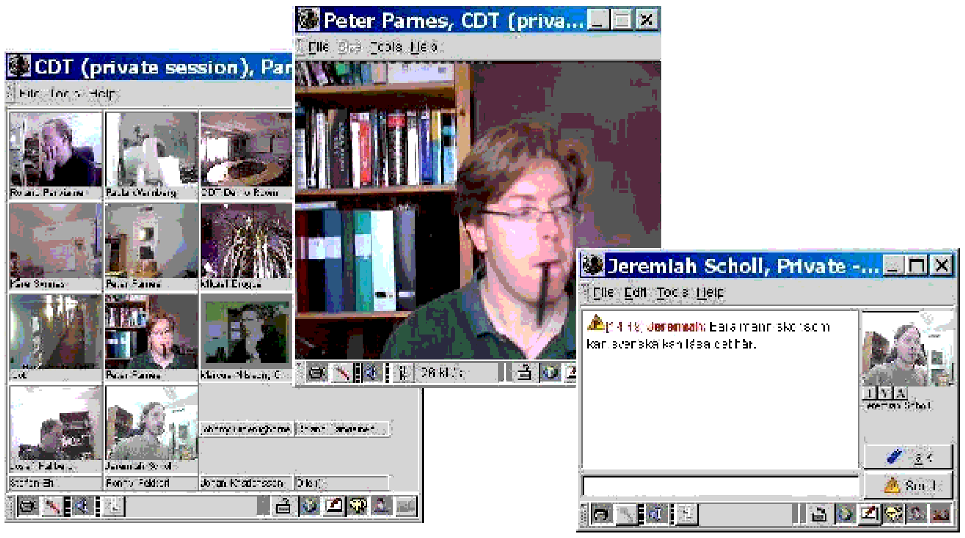
CDT (private session), Par
Peter Parnes, CDT (priva...
Jeremiah Scholl, Private -...
File Edit Tools Help

Figure 5.1: Video windows included in the Marratech Work Environment.

application. Sending low-rate video reliably but high-rate at best effort is just one example of the use of application semantics. In [3] Elf and Parnes presented a framework primarily aimed at relaxing reliability for efficiency and error handling reasons using application semantics. This approach is applicable also for the present work and though not being a part of the present prototype experiment, its application to bandwidth adaptation schemes and our collected effort relating to cost-efficiency in bandwidth use is highly relevant.

5.2 The Design of a Video Bandwidth Adaptation Scheme

The bandwidth sharing scheme described in this section follows the same architecture as SCUBA [1], but also differs from it in several ways. The first is that a novel approach for bandwidth sharing is used that seeks to first fulfill the minimal needs of all senders before dividing the remaining bandwidth among important group members. In addition, information about user interest is used to help each sender select the correct parameters in the tradeoff between image resolution and frame rate, which is something that SCUBA does not take into consideration. Another key difference is that optimizations for message passing are presented in the context of empirical observations made about how humans interact with collaborative workspaces, whereas SCUBA presented an alternative method based on statistical sampling. Finally, greater flexibility is shown in the number and types of user-behavior explored.

In order to determine the video streams that are of interest to a particular receiver one must answer the question, *"Who is this user currently viewing, and in what context?"*

Table 5.1: Video windows in Marratech client.

Window	Purpose of Panel	Pixel Resolution	Min. Frame Rate
Focus	high level of presence	702 x 576, 352 x 288, 176 x 144	5 fps
Private Chat	response clues	176 x 144, 88 x 72	1 fps
Participants	overview of activity	88 x 72	.2 fps

In regards to viewing context, the video windows provided inside the application user interface define the range of possible answers. The Marratech Work Environment [7], which we have used for prototyping, is shown in Figure 5.1 and includes video panels in several different windows. These windows are designed to complement each other and allow each participant to view other members in a variety of ways. Similar to the well known research application vic [9], Marratech provides users with a "Participants" window, which gives a thumbnail overview of the video streams currently received from the group, and a "Focus" window that displays the video obtained from a single group member at higher resolution. In addition to these windows, Marratech also contains a small video panel in each private chat display, which allows two participants to easily obtain response clues such as posturing and facial gestures (smiling etc.) while chatting in private.

Table 5.1 lists each of these windows specific roles in presence delivery along with their individual resource requirements including resolution and minimum ”acceptable” frame rate, which was obtained from a variety of sources. In regards to the Focus window, the minimum acceptable rate is derived from the various work summarized by Chen [2]. This includes work by Tang et al. [12], who noted that users consider 5 fps to be tolerable, and Watson et al. [14], who found that users do not perceive audio and video to be synchronized at frame rates lower than 5 fps. Other studies have also shown little difference in communication behavior or task outcome between 5 fps and 25 fps [5, 8, 4]. Together this work suggests that 5 fps will provide users with an adequate experience in a variety of situations requiring a high amount of attention. The values provided for the Focus window and private chat windows were obtained through a survey conducted of expert Marratech users, and reflect the values most typically reported as ”tolerable” for the each of the respective windows.

5.2.1 Identifying Important Video-Streams

The primary method for detecting user interest is to monitor user interface parameters that will reveal the video senders currently loaded in each of the video panels described above. In our case, this leads to a host giving one of four possible classifications to each sender, one for each of the separate video window configurations and one classification for members that are currently not viewed in any available panel. For some applications it may also be desirable to create classifications that describe senders contained in multiple panels simultaneously, but with the Marratech environment this is not necessary because the frame rate and resolution required for panels delivering a high level of presence will also be sufficient for each lower level. Thus, a video stream that is delivered for the Focus window will also be sufficient for the Participants window and so on.

Cross-media clues can also be used to detect an important video stream [1] with the most useful example being the monitoring of audio. The current audio sender is usually a leading presenter or an otherwise important participant in group discussions so the Marratech application gives users the option of selecting ”video follows audio”, which will automatically move the current speaker into the Focus window. Monitoring the content of the Focus window will still be sufficient to detect an important stream in this case, but the audio clue can be useful to reduce the latency it takes for a sender to realize its importance and to further prioritize audio senders over other ”focused” participants as described in the next subsection. This is also discussed in section 5.2.4 in relation to the evaluation of the prototype behavior.

The whiteboard and chat can also provide useful clues, but of a somewhat different nature than audio and video. While drawing with the whiteboard pen or sending a chat message may be a sign that a user has become interesting to other users, this will likely only be for a short period of time while they ”check out” the user’s activity. Therefore, when a sender has a low frame rate (less than 1 fps) an event from either of these media can be used in order to have him send an extra frame or two.

Downgrading a Sender

At times user interface monitoring and cross-media clues can be misleading and may cause a client to identify senders as important when in fact their video feeds are expendable. Electronic corridor participants can for example typically leave their office for an extended period of time, which may result in a client that continues to act on the behalf of its user even though no one is in the room to view the video streams received. One strategy that can be adopted in order to minimize the impact from this type of misidentification is to obtain hints regarding events external to the application before making decisions on behalf of the client. Hints of this type work to downgrade a sender that would otherwise be identified as important, and can further refine the process of detecting user importance. Several example hints in this category are listed below.

Detecting Idle Receivers It is pointless for a receiver to continue requesting video from senders when no one is actively using the computer. One primary method for detecting an idle receiver is to monitor the user’s screen saver. This can be complemented by other techniques, such as the monitoring of peripheral input devices like the keyboard and mouse, and/or the detection of a lack of movement in front of the user’s camera.

Window placement When windows from other applications cover up a video panel, it is a solid indication that the user is not interested in the incoming video stream [6]. This should also be true if the video window in question is minimized.

Limited Resources Even if a user can benefit from receiving additional data it does not guarantee that he has enough resources to do so. This can be especially true when using a mobile client as they are often more limited by CPU and memory resources than available bandwidth.

Table 5.2: Estimated bandwidth usage for each sender.

Window	Frame Rate	Resolution	Bandwidth Usage
Focus	5 fps	352 x 288	55 kb/s - 160 kb/s
Private Chat	1 fps	88 x 72	8 kb/s - 20 kb/s
Participants	.2 fps	88 x 72	less than 1 kb/s - 4 kb/s

5.2.2 Video Adjustment Algorithm

The video adjustment algorithm we have designed works by first to provide each sender with the minimum acceptable frame rate and proper image resolution for its most interested receiver, with unused bandwidth beyond that point distributed evenly among the highest priority senders in the group. The rationale for using this ”minimum requirements first” strategy is that it allows important senders to deliver the richest experience possible while keeping them from punishing less important senders. The main drawback of this method is that it may not be appropriate for use with sessions that have very limited bandwidth, for example those which intend to support modem users, because the aggregate requirements of even the least demanding senders may be hard to meet. However, sessions of this type are today generally viewed as a special case and most likely need a scheme that is optimized specifically for use with low bandwidth sessions [2], rather than a scheme that is designed for general use like the scheme that is described in this paper.

Obviously, it is not realistic to assume that the minimum acceptable requirements will be the same in every situation. In practice the administrator of the session should have the option of setting these values. However, in order to make the creation of sessions more user friendly it is important to have a workable set of default values that can be used when the administrator does not exercise this option. With this in mind we have done an analysis of expected bandwidth usage when sending at several of the appropriate frame rates and image resolutions discussed in Table 5.1.

Table 5.2 includes this information and can be used as a reference when trying to determine how well a minimum requirements approach will scale in the real world. The bandwidth measurements included were taken from a Marratech e-meeting client while sending video data at various frame rates and resolutions included in Table 5.1. The fourth column in Table 5.2 shows bandwidth measurements taken during ”typical” use, with the low value representative of users that are fairly still in front of their computer, and the high value taken during moments of high activity, such as the user moving about or interacting with another person in the office. It should be mentioned that although the private chat and Focus windows have variable resolutions our measurements were taken with the default settings applied.

The numbers in Table 5.2 should only be treated as estimates, as variations in bandwidth consumption can be expected due to real-world factors, such as the camera type in use and the amount of motion between frames. They do however show that for a typical session (less than 50 users) it is not difficult to meet the minimum requirements for the Participants window due to the low bandwidth required by each sender. In practice this is also true for private-chat users because the concurrent number of chats is usually equal to a small fraction of the number of session participants. However, the requirements of each ”more important” sender, defined as those currently sending audio or being viewed in the Focus window, may be difficult to meet if the attention of the group is too ”spread out” or if the session has low to medium available bandwidth (256 Kb/s - 500 Kb/s).

Each sender operates within the scheme by classifying itself on a scale from 0 to 4 based on how it is viewed by other group members and whether or not it is currently sending audio. These classifications are:

4 - audio sender

3 - Focus-window sender

2 - private-chat sender

1 - Participants-window sender

0 - no interested receivers

A host uses information about its class in order to determine its frame rate and resolution as given in Table 5.1, and measures the incoming bandwidth consumption of other members in order to determine the amount of bandwidth available to it. The sender then uses this information in order to adapt its video using the priority scheme described below.

Step 1: Bandwidth is divided evenly between all the senders until each sender can send at the minimum frame rate and resolution for the Participants window.

Step 2: If there is still session bandwidth available after step 1, it is allocated between the senders of class 2 or higher until they are sending at the minimal frame rate and resolution for the private chat window.

Step 3: If there is still available bandwidth after step 2, it is divided between senders of class 3 or or higher until they can send at the necessary frame rate and resolution for the Focus window. This is done first for class 4 senders, and then for class 3 senders.

Step 4: All remaining bandwidth is divided evenly between each sender in class 3 and 4.

The algorithm is more formally described in Figure 5.2. The function *availableBandwidth()* supplies the bandwidth that has been allocated as maximum for this session. This maximum value can be either a session parameter or obtained in some other fashion. The parameter applies to all members of the session, and when a bandwidth allocation algorithm is at work, the *availableBandwidth()* signals the practical upper limit.

This algorithm has also been implemented in the analyzer that is used to interpret the log data collected by the prototype application.

5.2.3 Receiver Feedback

In order for a sender to be aware of how it is viewed by other group members a mechanism needs to be in place that allows each receiving host to communicate their interests via mes-

```
$b_0 \leftarrow availableBandwidth()$

if $((r_1 \sum_{i=1}^{4} n_i) > b_0)$
    done

$s_{ij} \leftarrow b_1 \; \forall \; i, j$

$s_1 \leftarrow \sum_{i=1}^{4} \sum_{j=1}^{n_i} s_{ij}$

if $((b_0 - s_1) < r_2(n_2 + n_3 + n_4))$
    done

$s_{ij} \leftarrow s_{ij} + r_2 \; \forall \; i >= 2, j$

$s_2 \leftarrow \sum_{i=1}^{4} \sum_{j=1}^{n_i} s_{ij}$

if $((b_0 - s_2) < r_3 n_4)$
    done

$s_{4j} \leftarrow s_{4j} + r_3$

$s_3 \leftarrow \sum_{i=1}^{4} \sum_{j=1}^{n_i} s_{ij}$

if $((b_0 - s_3) < r_3 n_3)$
    done

$s_{3j} \leftarrow s_{3j} + r_3$

$s_4 \leftarrow \sum_{i=1}^{4} \sum_{j=1}^{n_i} s_{ij}$

$b_4 \leftarrow (b_0 - s_4)$

if $(b_4 \leq 0)$
    done

$s_{ij} \leftarrow s_{ij} + b_4/(n_3 + n_4) \; \forall \; i \geq 3, j$
```

Where,

b_0	Originally available bandwidth
$b_{1..4}$	Consumed bandwidth
$r_{1..4}$	Minimum rates for class 1..4, acc. to Table 5.2
s_i	Total bandwidth for class i senders
i	$1..4$ is the bandwidth classes
j	$1..n_i$ is the number of senders in class i

Figure 5.2: Pseudo code describing the bandwidth allocation algorithm.

Table 5.3: User interactions logged during empirical study and influence on sender class.

Event	Bandwidth class	Up-/Down-grading
Un-muting audio	4	Up
Muting audio	4	Down
Viewing or maximizing video	3	Up
Minimizing or closing video	3	Down
Opening private media	2	Up
Closing private media	2	Down
Un-muting participant video	1	Up
Muting participant video	1	Down

sages. The simplest way to do this is to have each receiver automatically send a message each time an event occurs that causes it to reclassify a sender. This approach may of course end up in unnecessary messages being passed but it is not clear if this will consume enough bandwidth to significantly reduce the performance of the application.

Several techniques can be applied in order to reduce the number of unnecessary messages with the most obvious example occurring when someone starts to send audio, which will cause them to be moved into the Focus window by several participants simultaneously. In this situation a more efficient approach then having each receiver send a message is to instead have each receiver inform the group when they change the ”video follows audio” option, which will enable the accurate use of the audio clue mentioned in sect. 5.2.1.

An unnecessary message may also be created when a receiver views a sender in a new context while already receiving enough video. For example, if a sender has a frame rate of 5 fps due to the actions of other receivers, it is pointless to send it a message when opening up a private chat window, as this requires a refresh rate of only 1 fps. The number of messages of this type can be reduced by having each receiver monitor the frame rate and resolution of incoming streams and pass messages only when they are deemed to be inadequate. The drawback of this technique is that it will make it difficult for senders to know exactly whom every receiver is watching, and will thus require an additional mechanism so that each sender can find out when they should reduce their bandwidth after receivers have lost interest in them. A simple way to handle this is to include information about how each sender is viewed in RTCP receiver reports, which will solve the problem, but will also introduce latency in the bandwidth reduction process. SCUBA takes a different approach towards feedback and uses statistical sampling rather than obtaining messages from the entire group. The advantage of this method is that it improves overall scalability because the number of messages grows logarithmically rather than linearly as the session size increases.

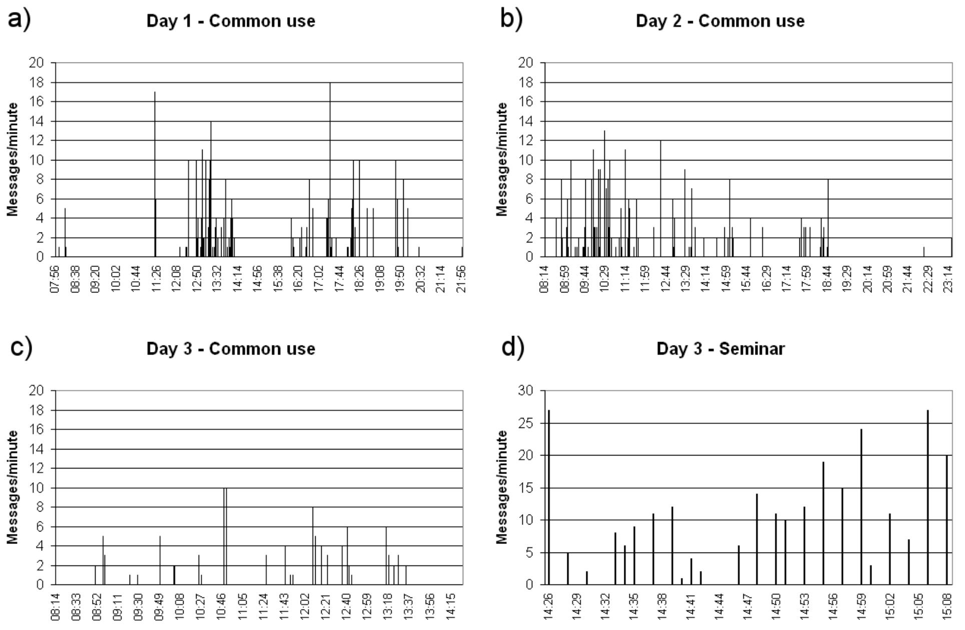

Figure 5.3: Graphs showing the total number of interactions within a 9-user research group over a three-day period. (Note the scale difference in (d).)

5.2.4 Prototype experiences

Because messages should only be created based on specific user interactions it is not clear if any of the above message reduction strategies are necessary, or if these interactions will typically be infrequent enough to make the number of messages passed in the session negligible. In order to gain further understanding of the potential amount of bandwidth that messages may consume, we conducted an empirical study of a research group consisting of nine daily Marratech users. This was done by creating a prototype version of Marratech, which generates messages based on specific user interactions as summarized in Table 5.3, and distributing it among these users. The messages were logged over a three-day period under normal working conditions, which included a formal research discussion on the last day, as well as periods of more ”common” use.

Figure 5.3 shows four graphs of activity during the logging period, in which a total of 1046 interactions were detected, corresponding to 119244 bytes worth of data. The average message length was 114 bytes, which included a sender id, timestamp and an indication of the interaction in question. It should be noted that these messages were not optimized in any way, so in practice it should be possible to reduce this size. Graphs **a**, **b** and **c** show activity during ”common use” periods and are highlighted by a fairly low amount of activity, with some short bursts occurring that correspond to increased interaction between the users.

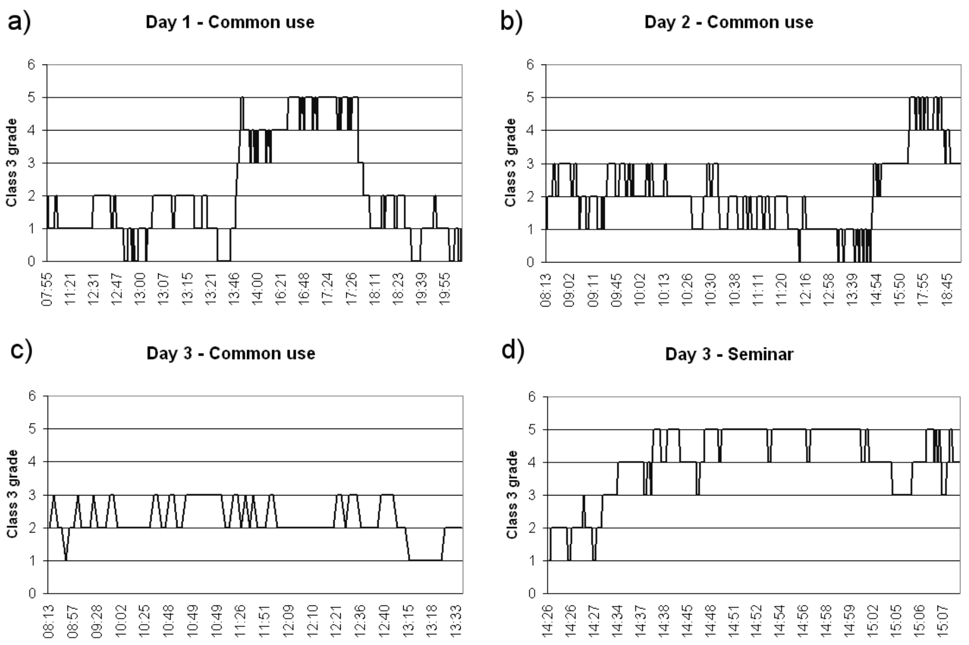

Figure 5.4: Graphs showing the total grading of senders based on receiver’s interest in their video within a 9-user research group over a three-day period, evaluated from the log data described in Figure 5.3.

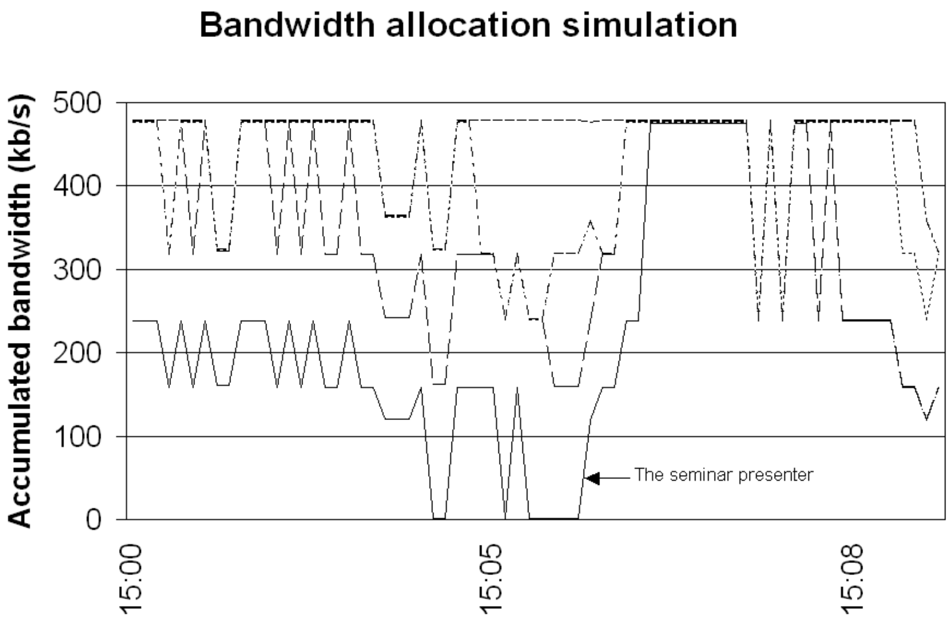

Figure 5.5: Graph showing the simulated cumulative bandwidth for selected senders from a portion of the seminar scenario, evaluated from the log data described in Figure 5.3. As is evident from the graph, the session bandwidth limit is 500 kb/s.

As expected the most intensive period of message creation by far occurred during the research discussion on the last day, which is shown in graph **d**. This included the hour of highest activity during the three days, in which 380 messages were sent. The peak minute of usage during the research discussion resulted in a total of 27 interactions, which corresponds to an average bandwidth consumption of less than .05 kb/s and a total bandwidth consumption of 3068 bytes. This shows that even if all the messages during the most active minute of the discussion were created simultaneously that the amount of bandwidth consumed would be negligible.

Thus, a conclusion can be drawn from this study that during normal use the total number of messages expected should consume a tiny portion of the session bandwidth, even if no optimizations are in place. Thus there is a fair amount of latitude here, before the messaging possibly becomes a problem. This includes radical optimization of the message format, and piggybacking messages onto other information packets.

Figure 5.4 shows four graphs demonstrating the sum of “gradings” computed by the log file analyzer implemented in perl for each log event during the logging period of three days. In this particular sense a sender is afforded one “grade” when one receiver views that sender in the main video window. Since one or more receivers can view one sender, each sender can well obtain a higher grade than one. More generally, if there are n members of a session, the highest possible grade is then $n - 1 - \sum g_i$, where g_i is the grade of an individual instance of the other session members. The maximum total number of grades would be n for the not so practical situation that everyone in the session is watching each other in a logic “circle”. Then, every session member would obtain an equal share of the bandwidth. Otherwise, since a sender cannot obtain a grade for watching its own video, the maximum grade is $n - 1$, for example in the lecture scenario where every receiver is paying attention to the lecturer, s/he would obtain most of the bandwidth, at least up to some practical limit.

In the group, consisting of 9 active people, that used our prototype, the highest grade was 5 of the practical limit 8.

It might be suggested that the bandwidth-allocating algorithm should take a sender’s grading into account, so that a sender in class 3 with grade 4 (designated grade 3.4) would be considered more important than a sender with grade 3.1. This would give the advantage for example in lecture scenarios, that people who occasionally divert from the lecturer to view someone else in the session, would not have an impact on the lecturer’s video bandwidth. The lecturer would also be “protected” by the fact that sending audio pushes the sender to class 4, and since class 4 will be allocated before class 3, this problem would be most pronounced during times when there is no audio.

From the figures (5.3 and 5.4) it is evident that there are substantial periods when the activity is very low and when no user is shown in someone’s main video window, the Focus window. This is fully consistent with the “electronic corridor” scenario, which was referenced earlier in Section 5.1. The bandwidth allocated according to the presented scheme during such periods is very low, amounting to 1 kb/s for each participant. There may be a significant amount of “idling time”. We suggest that this time be put to better use, by sending the 0.2 frames per second (cf. Table 5.2) reliable to reach a higher level of quality which would be especially beneficial in relation to the very low frame rate. It has been shown e.g. by Chen [2] that this is a feasible solution.

Due to the low frame rate this would hardly pose a timing problem and it would supply the viewer of the electronic corridor with a picture that is always correct.

The same analyzer that was implemented to parse the logged information for user activity and user grading also implemented the bandwidth-sharing algorithm presented in Figure 5.2. The algorithm makes use of a set of constraints, which are the session bandwidth limit and the target rates for the different classes of senders (cf. Table 5.2).

This implementation of the log file analyzer operated on the log records only and gave a simulation of what bandwidths the proposed algorithm would have allocated to the different users. A small part of the result is presented in Figure 5.5. The figure shows, for a duration of only 10 minutes, the proposed bandwidth allocations for the 5 most active senders. The curves are accumulative and apart from the small amount of bandwidth represented by the removed low-rate senders, the sum of the bandwidths always amount to the session limit, which in this run was set to 500 kb/s. The graph shows rapid bandwidth fluctuations and that two or possibly three people are watching each other’s Focus windows towards the end of the seminar discussion.

It stands to reason that when implementing this algorithm into Marratech Pro, there will have to be additional constraints, most notably timers to prevent the bandwidth to change too often. Even if the previous experiments show that the messaging traffic does not constitute a problem *per se* there will be little sense in changing the bandwidth for a user in extremely short intervals, resembling for example the round trip time between sender and receiver.

The experiments also show that the allocation algorithm should be expanded with the notion of sender grading.

5.3 Summary and Conclusions

We have introduced a framework for bandwidth sharing in video conferencing that uses the implicit detection of user interest as a metric for resource allocation. Schemes of this type contain three architectural components, which are the detection of user behavior, message passing, and bandwidth adjustment algorithms. In the area of user-interest detection we have described several methods for identifying users' interests and have introduced new ways to reduce the number of false positives in this process. In addition, we have expanded the area of bandwidth adjustment in order to help senders correctly identify their optimal frame rate and image resolution in each situation and have done so by adopting a ”minimum requirements first” strategy. This strategy attempts to provide each sender with the minimum frame rate and image resolution for its most interested receiver before assigning the remaining session bandwidth to the senders deemed to be most important.

We have also discussed several different mechanisms designed to reduce the number messages created, and have conducted an empirical study in order to determine how necessary they are during real use. This study was conducted by deploying a prototype we created among a research group at our university that allowed us to monitor the messages they generated by their behavior using the collaboration application. We concluded from this study that, given the interactions from Table 5.3, messages will occur infrequently enough during normal use that such message reduction mechanisms are of little use in practice, even though they may be academically interesting. From further analyzes of the user behavior logs, including the grading of senders and the simulation of bandwidth allocation based on the user behavior, we concluded that grading of senders can be a valuable tool and that the bandwidth allocation scheme must smooth out allocations over time.

In the introduction 5.1.1 we presented three research questions that represent the driving force for this work. These questions have been addressed in this paper as follows.

Application design for cost-effective media distribution. In general we mean that in order to cost-effectively distribute media streams under varying usage scenarios and in heterogeneous environments, applications must be designed to take into account not only error handling in relation to application semantics, but also as we propose, handle *user behavior*. It is our opinion that a framework like the one proposed in this paper would enhance the success rate of building this kind of design into applications.

On the design of dynamic video bandwidth application schemes. A scheme for dynamic video bandwidth allocation that is designed based on these conclusions, would help applications establish a video presence using the suggested ”minimum requirements first” strategy and would furthermore cater for conservation of bandwidth by allocating resources only to those senders who, according to other listeners behavior, are the most relevant resource consumers.

Handling widely varying usage scenarios. Handling the widely varying use-cases or scenarios in which collaborative applications are used is indeed a problem when it is acted upon at a “protocol” level. Our approach to act at the topmost level, the user behavior, helps us to handle extremely different scenarios equally effective.

5.3.1 Future Work

Our first priority in the future is to make a complete, user-friendly prototype that can be distributed among the Marratech test users. This will require us to look into several issues including user-interface options so that users can ”opt out” of the dynamic bandwidth process. In the real world this will be necessary because there are some situations when it is most beneficial to allow certain users in the session to set their bandwidth consumption manually. We also plan to study the performance of our scheme in this type of mixed environment.

Furthermore, as this kind of on-line collaborative environment becomes more popular, it may become common that users participate in more than one session at the same time. In the research group at Luleå University of Technology, this is definitely already the case. There is obviously a possibility of conflict here since such a user will probably not be of equal interest to others in all sessions. Therefore, the user’s classification (class.grading) will not be relevant over different sessions and the user will “steal” bandwidth in the sessions where the interest is lower, and will “loose” bandwidth in sessions where the interest is higher. Presently, Marratech does not support sending different-rate video streams to separate sessions and will use the “worst case” strategy and thus use the video rate of the session with the highest constraints and, therefore, the lowest frame rate. We will investigate these implications in future work. Possible solutions include allowing for different frame rates in different sessions.

In addition, robust user studies are needed in order to find further ways of refining the bandwidth allocation scheme. In particular, it is not clear at this time if it is best to divide all the extra bandwidth between only the important senders in the group as stated in sec. 5.2.2, or if there is a more optimal strategy. In some situations for example it may be better for a portion of the extra bandwidth to be used in order to increase the frame rate of clients in the Participants window.

Acknowledgments

This work is supported the Centre for Distance-spanning Technology, the Swedish Research Institute for Information Technology, the VITAL project and the Mäkitalo Research Centre.

Stefan Elf is also with Ericsson AB, SE-931 87 Skellefteå, Sweden. Views expressed in this paper are his own and not necessarily shared by his employer.

Bibliography

[1] E. Amir, S. McCanne, and R. H. Katz. Receiver-driven bandwidth adaptation for light-weight sessions. In *ACM Multimedia*, pages 415–426, 1997.

[2] M. Chen. Achieving effective floor control with a low-bandwidth gesture-sensitive videoconferencing system. In *ACM Multimedia*, 2002.

[3] S. Elf and P. Parnes. Applying semantic reliability concepts to multicast information messaging in wireless networks. In *IRMA Conference Proceedings: Issues & Trends of Information Technology Management in Contemporary Organizations*. Information Resources Management Association, Idea Publishing Group, May 2002.

[4] G. Ghinea and J. Thomas. Qos impact on user perception and understanding of multi-media video clips. In *ACM Multimedia*, 1998.

[5] M. Jackson, A. H. Anderson, R. McEwan, and J. Mullin. Impact of video frame rate on communicative behaviour in two and four party groups. In *ACM Computer Supported Cooperative Work*, pages 11 – 20, 2000.

[6] W. Kulju and H. Lutfiyya. Design and implementation of an application layer protocol for reducing udp traffic based on user hints and policies. In *5th IFIP/IEEE International Conference on Management of Multimedia Networks and Services, MMNS*, 2002.

[7] Marratech. - The e-meeting company. URL[1], March 2003. Visited March 30th, 2003.

[8] M. Masoodian and M. Apperly. Video support for shared work-space interaction: An empirical study. *Interacting with Computers*, 7(3):237–253, 1995.

[9] S. McCanne and V. Jacobson. vic : A flexible framework for packet video. In *ACM Multimedia*, pages 511–522, 1995.

[10] M. Ott, G. Michelitsch, D. Reininger, and G. Welling. An architecture for adaptive qos and its application to multimedia systems design. *Special Issue of Computer Communications on Guiding Quality of Service into Distributed Systems*, 1997.

[11] J. Scholl, S. Elf, and P. Parnes. Efficient workspaces through semantic reliability. In *10th International Conference on Telecommunications, ICT*, 2003.

[1] <http://www.marratech.com>

[12] J. C. Tang and E. A. Isaacs. Why do users like video? studies of multimedia-supported collaboration. *Computer-Supported Cooperative Work: An International Journal*, pages 163–196, 1993.

[13] J. Wang and D. G. Frank. Cross-cultural communication: Implications for effective information services in academic libraries. *portal: Libraries and the Academy*, 2(2):207–216, April 2002.

[14] A. Watson and M. A. Sasse. Evaluating audio and video quality in low-cost multimedia conferencing systems. *Interacting with Computers*, 8(3):255–275, 1996.