mWeb: a framework for distributed presentations using the WWW and the MBone*

Peter Parnes, Mattias Mattsson, Kåre Synnes, Dick Schefström
Department of Computer Science/Centre for Distance-spanning Technology
Luleå University, Sweden
{Peter.Parnes,Mattias.Mattsson,Kare.Synnes,Dick.Schefstrom}@cdt.luth.se

Abstract

This paper presents a framework for bringing the MBone and the World-Wide Web closer together by real-time distribution of HTML-pages and synchronization of WWW-browsers. This framework will allow for distributed presentations with WWW-material using the mWeb application. Distribution is done using two new protocols, Scalable Reliable Real-time Transport Protocol - SRRTP and Scalable Reliable File Distribution Protocol - SRFDP. Synchronization is achieved using the WebDesk Control Bus which allows for messaging in large groups of programs and hosts. This paper also presents how the mWeb application can be used for programmed presentations resulting in a television-like Web-TV.

1 Introduction

The World-Wide Web (WWW) is today mostly used as a request application where the end-user tells a browser to fetch a page which is then displayed. Imagine instead your WWW-browser being used as a broadcast[1] receiver where the user tunes into a channel and starts watching the HTML-pages being distributed there. This would add a real-time dimension to the WWW that could be used for *distributed presentations* and so called *programmed presentations, (Web-TV)*

The until now experimental Multicast Backbone (MBone) [5] is today being deployed on the Internet as part of the real production network and more and more sites around the world are being connected each day. The MBone is daily being used for both one-way broadcasts of many different events as concerts, conferences, lectures and tutorials and for multi-way sessions as distributed meetings, virtual worlds, games, multiuser chat/talk and computerized surveillance. Although many tools exists, there is a lack of tools for scalable distribution of documents, such as WWW-pages, and synchronization of programs, such as WWW-browsers.

Many documents exists on the WWW today and many more are produced every day and it seems therefore very natural that these documents should be used in distributed presentations. The realization of a good tool for this purpose would add a new dimension, a *real-time dimension*, to the WWW.

The problem of creating a framework for using the WWW as a distributed presentation medium can be divided into two parts, a *synchronization* part and a *distribution* part. *Synchronization* relates to how to make all WWW-browsers within one session display the same page at the same time. *Distribution* relates to how the HTML-pages are distributed to all listeners in a scalable way.

The first part is solved by using a new Control Bus (CB) protocol [1] which is a messaging protocol based on a reliable multicast framework called *Scalable Reliable Real-time Transport Protocol (SRRTP)* [12]. SRRTP is an extended version of the Real-time Transfer Protocol (RTP) [10] to include the ideas from the SRM framework [11].

A simple solution to the second part would be to let each client fetch the data to be presented using the standard way of fetching WWW documents, but if the group is large this would create a very large burden and high latency on the HTTP-server serving that particular page. This simple solution doesn't scale to many members. Instead, a solution using a new protocol, *Scalable Reliable File Distribution Protocol - SRFDP*, for distribution of not only files but also meta-information about these files, is presented.

This paper also presents an application called *mWeb* that can be used for simple collection of presentation material, distribution of this material, synchronization of WWW browsers and a new concept,

*This work was supported by the Centre for Distance-spanning Technology (CDT), Luleå, Sweden.

[1]broadcast in the comparison to TV-program broadcasts and not network-broadcasts.

called *programmed presentations* or *Web-TV*.

1.1 Related Work

At least two earlier attempts have been made on the Internet to use the WWW as a distributed and synchronized presentation medium:

In 1995, Ed Burns presented his work on *Web-Cast* [13], a platform for sharing WWW documents over the MBone by either multicasting URLs or HTML-documents. It interfaced the WWW browser XMosaic [15] through the Common Client Interface (CCI) [2] and distributed WWW pages and corresponding inline images using the Reliable Multicast Protocol (RMP) [9]. Tests were conducted during the spring of 1995 which showed that the multicast distribution protocol used by that version of RMP wasn't suitable for wide-area-networks, because only the original sender of the data could do a repair of lost packets[2].

mMosaic [7] is another tool for sharing WWW-documents over the MBone. It's currently being developed and is an extended version of the WWW browser, XMosaic. Initial tests show that mMosaic works well with HTML-pages and smaller images, but the distribution delay gets to large with bigger images. A drawback, is that it's very tightly coupled to XMosaic.

Both these tools and other earlier Web-distribution tools are all tightly connected to the XMosaic browser. One of the main design-constraints on the mWeb application is that it should be browser independent.

Real-time Transfer Protocol (RTP)

The new protocols presented in this paper are based on the Real-time Transfer Protocol and the framework of SRM.

The Real-time Transfer Protocol (RTP) [10] is designed for real-time traffic over both Intra- and Internets. RTP functions include loss detection for quality estimation and rate adaption, sequencing of data, intra- and inter-media synchronization, source identification and basic membership information. RTP is designed to operate on any kind of network-protocol and is therefore completely self-contained meaning that it doesn't depend on any information in the lower levels of the network-model. Although, today only implementations of RTP over IP/UDP exists, it can for instance be used over native ATM or ISDN.

RTP actually consists of two different protocols, the RTP that transports the data itself, called payload, and the Real-time Transfer Control Protocol (RTCP), which is used for exchanging meta-information about the current session. These protocols operate on two different channels and all traffic is sent in a best-effort way meaning that packets can be lost without retransmission. On the RTCP-channel all members send out regular messages about how much data they have sent (if any) and how well the data on the RTP-channel is being received. This information can be used to detect packet-loss and request the senders to adapt their transmission-rates to the current packet-loss.

Scalable Reliable Multicast - SRM

Scalable Reliable Multicast (SRM) [11] is a reliable multicast framework for application level framing and light-weight sessions. The algorithms of this framework are efficient, robust and scale well to both very large networks and very large sessions. The framework has been prototyped in *wb*, a distributed whiteboard application, and has been extensively tested on a global scale with sessions ranging from a few to more than 1000 participants. The framework can be summarized as follows: There are basically three types of packets: heartbeats, NACKs and repairs.

Each member periodically sends out a *heartbeat* including the sequence number of the latest packet it has sent. These heartbeats are used by the other members to detect packet loss by comparing the sequence number in the heartbeat and the sequence number of the last data-packet received.

If a packet is lost a negative *acknowledgment (NACK)* is sent to all members using the same method of transportation as the original data. As all members see these NACKs, everyone can participate in the repair of the traffic. This is important because it eases the burden on the original sender and makes the repair process faster if the distance between the NACK sender and the original sender is large.

Each member of the session cache the latest data-packets received (and sent) and if they see a NACK for a packet they have in their cache, they retransmit that packet to the whole group as a *repair*.

To minimize the number of NACKs and repairs sent these two operations are preceded by *exponential back-off* meaning that instead of sending the packet directly, the client waits for a random time (based on the distance to the related other party) and if another client sends a corresponding packet it withdraws its own potential transmission.

2 Scalable Reliable Real-time Transport Protocol - SRRTP

In [12] a proposed protocol, called *Scalable Reliable Real-time Transport Protocol - SRRTP*[3], is presented.

[2] In more recent versions of RMP the distribution problems have been solved.

[3] This protocol was earlier called RTP/SRM.

This protocol incorporates the framework from SRM into RTP by extending both the RTP and the RTCP protocols. This provides a reliable and scalable protocol for among other things wide-area file-distribution and messaging between applications. (Note that a earlier version, called DVRMP, of this protocol was presented in [3].)

All pseudo-random timers and back-off times in SRM are based on the "distance" to the originator or destination. The distance is calculated using the heartbeats and the round-trip-time it takes for a packet to travel between two members.

The added control messages (heartbeats, NACKs, time-stamp queries and replies (data needed for distance-calculations) are sent using a new RTCP packet type. Each packet in a session is uniquely identified by the senders SSRC (a RTP-header field which is used to uniquely identify a member within a group) and the packet-sequence-number. This packet-id is used when sending NACKs and if several packets are lost from the same source, several NACKs for all the lost packets can be grouped together into the same NACK-request to minimize the number of NACKs.

3 Scalable Reliable File Distribution Protocol - SRFDP

SRRTP provides a scalable and reliable framework for distribution of data within a large group of members. Because SRRTP is based on packets, there is a need for a protocol for division of files into packets and control of file-distributions. This sections presents a new protocol called *Scalable Reliable File Distribution Protocol - SRFDP*, which is a protocol for distribution of not only single files but also groups of files and meta-information about these files.

3.1 File-distribution

Each file-distribution process consists of three steps: deciding the *meta-file-information*, *sending the header* and *sending the file*.

The *meta-file-information* consists of information about a particular file. This information include the file-name, file-size, MIME-type [6] and the last-modification time. The last field is used by receivers to check if a locally cached copy of the file should be invalidated. This information is also referred to as the *header*.

The second step is to send the *file-header* to notify the members of the group of the upcoming file-distribution and finally the file itself is distributed using SRRTP.

Several headers can be grouped into an special *index-file*, which is used by clients to identify the files available from one sender within the session. Index-files also contains information about current file-distributions. Several index-files can be available within one session if several senders are offering files.

A typical client-scenario would look like this; the client joins the session and requests the index-file. If he doesn't receive any index-file (no one is offering files), he just waits until an index-file is offered. After receiving at least one index-file, he selects the files of interest and requests that these files will be distributed. Before any request, he applies exponential back-off to check if no-one else is also requesting the same files. For instance, if the session was just announced several clients might join the session and at the same time requesting the same information. He also checks the current traffic-flow for any active file-distributions. If a distribution of a file he is interested in exists, he saves the part of the file currently being distributed and when the end of the file is reached he requests the missing parts as if they where lost.

3.2 SRFDP and SRRTP

In SRRTP NACKs, were based on the packet-id but SRFDP extends this to allow NACKs based on a file-name and a byte-offset within that file. This can be used by a client to request a particular part of a file, without having to know the the current packet size being used.

Furthermore, the time-stamp in the RTP-packet header is constant (the time of when the first packet is sent is used) on all packets related to the same file distribution. This allows for several files to be distributed from one sender at the same time within one session.

4 The mWeb application

The technology represented by SRRTP and SRFDP have been deployed in an application called *mWeb*. This application includes functionality for distribution of HTML-pages including in-line data and embedded objects, pre-caching of files to be used within a session, on-demand fetching of files, synchronization between browsers and interfacing different WWW-browsers.

As described in the introduction the problem of adding real-time distribution of HTML to the WWW could be divided into two parts, *synchronization* and *distribution*. This section discuss the architecture of the mWeb application and how these problems have been solved in mWeb.

4.1 The architecture

mWeb acts as a gateway between a WWW browser and the MBone, mediating distribution of HTML-pages and *display-messages* (see section 4.2). The

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

1. URLs to be displayed including URLs to any in-line data are specified manually in a file by the presenter. This file is used by mWeb.

2. URLs are collected dynamically during a presentation using a browser that supports the Common Client Interface (CCI) [2] (today only the browser XMosaic supports CCI). This means that whenever the presenter selects a link or change HTML-page (for instance using the history in the browser) information is sent from the browser to the mWeb application. (The same functionality can be achieved using the *remote_netscape* program with the X11 Netscape Navigator browser, although the method is not as elegant as the CCI-based one).

3. URLs are collected dynamically during a presentation using the special *mWeb WWW-proxy* that sends information about the requested pages to the mWeb application. This is achieved by telling the browser to request all pages through the proxy, instead of fetching them directly.

The last two methods can also run in so called *cache-mode*, meaning that the presenter makes the collection of URLs to a file, before the actual presentation is held. This file can then be edited by the presenter to add or remove URLs. This is useful if a page to be displayed contains an embedded object, such as a Java-applet (see section 4.4).

When the presenter is finished with his page-list, he tells mWeb to start distributing the pages using SRFDP. Each listening client will cache the files locally and wait for the presentation to start. On user request, information about the current transfers can be displayed.

During the presentation a pop-up-window containing a list of displayed pages is shown. At the presenters side this list will contain *all* the pages but on the listeners side only pages that have already been displayed are listed.

4.2 Synchronization of WWW browsers

When a presentation is distributed over the MBone and a WWW-browser is used for presenting the slides, there is a need for synchronization between the involved WWW-browsers, meaning that all browsers display the same page. This is solved by sending a *display-message* to all members of the group using the *WebDesk Control Bus (CB)* [1]. The CB protocol is a messaging protocol used above SRRTP and it provides a simple way of exchanging messages both internally within an application and between several distinct applications.

During the session the listener can choose what shall happen when a new display-message is received: display the new page directly, display a dialogue asking the user what to do or just display in the page-list that a new display-message have been received.

If a listener wants to go back and take a look on an already displayed page, he can do that by selecting the page of interest in the list over displayed pages. He can also instruct the local mWeb client to send a display-message to all other listeners including the presenter. This is useful if the listener wants to comment or ask a question related to a page that is not currently displayed.

4.3 Interfacing WWW browsers

mWeb saves all received pages and data on local disk and if the name of the file contains one or more '/'-characters, a relative directory-structure according to the name will be built. This allows for relative links to sub-directories to be resolved locally.

When a page is to be displayed, a fictive URL with 'file:' as the protocol-part pointing to the local cache is constructed and the browsers is instructed to display that URL. If a requested page is not found at a client during the session, mWeb will request that page.

4.4 Embedded and in-line objects

It is easy for a program to scan an HTML-page for references to inline images as these are stated clearly in the file, but it becomes more complicated if the page contains embedded objects, such as a Java-applet. A reference to an embedded object only points out the first element of what could be a complex program. In the Java-applet case, the applet can fetch more data in two ways:

Classes (program-parts of the applet) are fetched from the same source as the main-class (the one referenced in the HTML-page) so if the Java-applets main source is local disk, requests for additional classes will be serviced locally.

Other data can be referenced in any way the applet programmer sees fit, so special care has to be taken by the programmer on how the data is referenced if the applet is supposed to be copied to other sites using mWeb.

4.5 Programmed presentations and Web-TV

Although this paper focuses on distributed real-time presentations the mWeb application can be used in so called *programmed presentation mode* where the presenter instructs mWeb to send display-messages for certain URLs at certain times. This can be used to create a television-like system where the receiving browsers display a group of pages. We call this functionality *Web-TV*.

5 Implementation and Status

The mWeb application and its underlying protocols are all currently being implemented using the Java language [4].

WWW browsers currently supported are Netscape Navigator and NCSA XMosaic. Although the Java-language itself is platform independent there doesn't exist any platform independent way of interfacing WWW browsers. Further information can be found at [8].

6 Summary and Conclusions

This paper has described a new application, *the mWeb*, and a framework for scalable and reliable distribution of HTML-pages and their in-line data to a large audience. mWeb allows for easy creation of WWW presentations, scalable distribution, local caching, on-demand fetching of data and synchronization between World-Wide Web browsers using the *WebDesk Control Bus*. mWeb can also be used to create programmed presentations resulting in a *Web-TV* framework.

Distribution of the data to be displayed is done using two new network protocols, *Scalable Reliable Real-time Transport Protocol (SRRTP)* and *Scalable Reliable File Distribution Protocol (SRFDP)*. SRRTP creates a scalable and reliable distribution platform for real-time data by adding the ideas from the SRM-framework to the Real-time Transport Protocol (RTP). As SRRTP only supplies a platform for reliable distribution of data, SRFDP adds a framework for grouping of files and distribution of meta-information about these files. The meta-information includes at least information about the names and the MIME-types of the files.

These new protocols and the mWeb application brings the WWW and the real-time MBone closer together and makes the distance to a true real-time WWW smaller!

Acknowledgments

Thanks should go to R.P.C. Rodgers for interesting comments and all nice encouragement.

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

References

[1] P. Parnes, *Control Bus Specification*, 1996, <URL:http://www.cdt.luth.se/~peppar/docs/>

[2] *NCSA Mosaic Common Client Interface*, <URL: http://http://www.ncsa.uiuc.edu/SDG/Software/XMosaic/CCI/cci-spec.html>

[3] *Network Support for Scalable Distributed Multi-party Virtual Reality*, <URL:ftp://cs.ucl.ac.uk/darpa/dvr.ps>

[4] JavaSoft Inc., *Java*, Technical Report, 1996, <URL:http://www.javasoft.com/>

[5] *The MBone information web*, <URL:http://www.mbone.com/>

[6] N. Borenstein, N. Freed, *MIME (Multipurpose Internet Mail Extensions) Part One and Two*, IETF RFC1521 and RFC1522, September 1993

[7] *mMosaic Information*, <URL:http://sig.enst.fr/~dauphin/mMosaic/index.html>

[8] *The mWeb application*, <URL:http://www.cdt.luth.se/~peppar/progs/mWeb/>

[9] *Reliable Multicast Protocol*, <URL:http://research.ivv.nasa.gov/projects/RMP/RMP.html>

[10] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications - IETF RFC1889*, January 1996.

[11] S. Floyd, V. Jacobson, S. McCanne, C. Liu, L. Zhang, "A Reliable Multicast Framework for Light-weight Sessions and Application Framing", *ACM SIGCOMM'95*

[12] P. Parnes, *Scalable Reliable Real-time Transport Protocol - SRRTP*, work in progress, <URL: http://www.cdt.luth.se/~peppar/docs/>

[13] *WebCast - Collaborative document sharing via the MBONE*, <URL:http://www.ncsa.uiuc.edu/SDG/Software/XMosaic/CCI/webcast.html>

[14] P. Parnes, D. Schefström, K. Synnes, *WebDesk: the Collaboration Support in MATES*, First European Workshop on Global Engineering Network, 1996, <URL:http://www.cdt.luth.se/~peppar/docs/>

[15] *NCSA Mosaic for the X Window System*, <URL: http://www.ncsa.uiuc.edu/SDG/Software/XMosaic/>