mMOD: the multicast Media-on-Demand system[*]

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

March 6, 1997

Abstract

This paper presents the multicast Media-on-Demand system, mMOD, which is an on-demand system designed for recording and playback of not only audio and video but also other media that are being multicast on the Internet/MBone today. The system allows for users to request playback of recorded sessions containing MBone audio, video, whiteboard, NetTextEditor, mWeb (a distributed HTML presentation system) or any other UDP-multicast distributed session. It also allows for random access within these recordings. The system include a Web-interface for starting and controlling running sessions. To compensate for jitter and out-of-order packets, sessions distributed in RTP-format (RFC1889), can be reconstructed with new time-stamps.

1 Introduction

A video-on-demand system (VOD) is a system that serves a number of clients with audio and video on their request, and that are usually limited to these two media. Normally one channel is allocated per request and receiver and this leads to resources being tied up. There exist proposals [12] for using multicasting and time-slots for making VOD systems scale better.

This paper presents the *multicast Media-on-Demand system, mMOD*, which is a system for playing not only audio and video but numerous kinds of media that exist on the MBone [6] today.

mMOD was created as a response to a direct need in an undergraduate-course being transmitted over the MBone, where we saw a large potential in students being able to go back and watch whole lectures or just parts of a lecture again. This kind of functionality has been accomplished earlier using analog video-technology, where students could borrow video-tapes at the local library. Unfortunately, this does not allow for presentation of more than audio and video. It also leads to delay and scalability problems in the distribution of video-tapes.

The goal of the mMOD system was to produce a system where users could easily request playback of electronic lectures and meetings transmitted over the MBone and have the possibility for random access within these playbacks. The recording-possibilities should of course include audio/video but also any other distributed media used within the session. Another goal was that the content-provider should easily be able to provide indexes in the presentation. Yet another goal was that it should be easy for the content-provider to record a session and that the final system should be completely portable between UNIX and Windows95/NT4.

The rest of this short paper is divided into section 1.1 on related work, section 2 on the architecture and design of the mMOD system itself, section 3 on the current implementation and section 4 include the current status and a summary.

1.1 Related Work

This section presents three different tools for recording and playing MBone sessions.

The MBone VCR

The MBone VCR [7] is a VCR application for use on the Internet's Multicast Backbone (MBone). This application uses a VCR interface to control recording and playback of sessions sent over the MBone. The VCR stores these sessions by synchronizing different media streams based on information provided by RTP (the Real Time Transport Protocol) [13]. It also allows indexing and random access within recorded sessions. A command language enables limited programming features, e.g. to schedule recordings or playbacks at a later point of time.

This tool can be used to record and play single sessions on arbitrary channels but can not be controlled remotely. It also have problems with sessions using

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

version 2 of RTP (which is the currently recommended and used version).

The Interactive Multimedia Jukebox

The IMJ (Interactive Multimedia Jukebox) [3] is a Web-based interface for requesting and scheduling of cartoons and movies. Sessions can currently be played out on one out of three different channels, of which two only can be viewed locally at the server-site. Only one session can be viewed at a time per channel, and once a session is started the session can not be controlled via the Web-interface.

RTPPlay and RTPRecord

RTPPlay and RTPRecord is part of the RTPTools-package [14]. These simple command-line tools gives the functionality of saving and playing RTP-sessions. The tools are used in the Interactive Multimedia Jukebox (see above).

2 The mMOD system

The mMOD (multicast Media-on-Demand system) is a system for recording and playing MBone sessions to a number of receivers. The mMOD can be controlled using a command-line interface or a World-Wide Web based interface.

The mMOD consists of two separate programs, *the VCR* and *the Web-controller*.

2.1 The VCR

The VCR is a stand-alone program for recording IP-packets on either UDP- or RTP-level. Recording on UDP-level means that it only stores the received packets without trying to parse them or rearrange them in any way. This is useful if data of unknown format is transmitted. Recording on RTP-level means that the VCR parses the RTP-header of each incoming packet and checks for duplicates and out-of-order packets. The recorder can record data arriving either in unicast or multicast.

The VCR can of course playback these recordings on either UDP- or RTP-level. Playback on UDP-level means that an exact copy of the original traffic (same variance between packets, same duplicate packets and same lost packets) as seen by the recorder is achieved. If the session is played back on RTP-level, the player tries to be smart and send packets based on their original time-stamps. This solves the problem of having a transport path from the original sender with a high variance in the transport delay. The recorder also rearranges packets depending on their sequence-number in the RTP-header.

VCR options

The recorder and player has a number of options which make recording and playback easier[1]. Recording-options:

- *Source-host filtering*: Only record traffic from a specific host. Useful for selecting traffic from a specific member of a session or if RTCP[2]-feedback from other listeners is unwanted.
- *Wait*: Do not start recording RTCP traffic until the first data-packet has been received. This can be specified per medium and lets one medium be in charge of the session. This is very useful if a meeting is to be recorded where several users start transmitting video before the meeting starts but do not say anything. In that case you do not want the recorder to trigger on the video-stream but the audio-stream.
- *Sender-timeout*: Stop recording if no data has been received for a certain amount of time. This could be followed by a new *wait* or the recorder exits when all media in a session has timed-out.
- *Append*: Append incoming data to the specified media-files instead of erasing them before starting the recording. A special marker, the A-marker, is stored at the boundary between the recordings (see the marker-option for playback below). This could be used for recording several lectures/meetings in the same file.

Playback options:

- *Receiver-timeout*: Stop playback if we do not receive traffic (control or data) from any receiver. This is followed by an optional wait-period before the player exits. If traffic is received during the wait-period the playback is resumed.
- *Marker*: Start playback at a special point in the recording based on markers. This lets the user select where to start the playback (see section "Data format" on page 3 for more information about markers).
- *Time*: Start playback at a certain amount of time into the recording. Good for resuming an earlier playback.

[1]Of course, all of these options are not specific to mMOD, but can be found in some existing on-demand applications

[2]RTP traffic actually consists of a data- and a control-stream. The control-stream (RTCP) contains information about the traffic of the session and minimal membership-information.

- *Max silence*: Skip silent parts which are longer than a specified max silence. Silence means that no data is being sent. This can be specified for the whole session (every medium has to be silent) or let one medium control the other. This is useful when a session consists of e.g. audio and video of a meeting where the participants have taken a break without stopping the video-transmission.

Address-specification

The addresses to record from and play to can be specified in two ways; either by *mappings* or as a *SDP-file* (Session Description Protocol) [15] which is the way sessions are announced on the MBone today.

A mapping is a string consisting of the media, the address and port, e.g. `audio=224.3.4.1/4398`.

The SDP-file can be stored locally or on a WWW-server and be specified by an URL. The later is useful if the recorder is used together with the mSD (multicast Session Directory) [9], a WWW-based MBone session directory.

VCR communication

The mMOD runs as a completely distributed system with no information about each VCR stored on disk. Instead, running VCRs are located using multicast. Each VCR listens on a known group/port pair for messages. If a message is an *alive-message* each VCR sends a response (via the same multicast-group) containing information about that VCR. This information contains a unique identifier, the name of the session, the owner (the user who created the session), if it is playing or recording, the media involved and a control-port.

The VCR can be controlled in two ways; either by sending multicast-messages addressed to a VCR using the unique identifier, or by connecting to the VCR using a TCP-connection[3] to its control-port. The first method is used by the Web-controller.

A VCR understands the following commands; *stop* - stop the session and exit, *pause* - pause the recording/playback, *continue* - continue after a pause, *skip X* - jump X number of A-markers (see above) where X can be both positive and negative, *rewind* - jump to the beginning of the session and *jump Y* - jump Y milliseconds in the session where Y can be both positive and negative. Information about the current state (e.g. if it is playing and where in the session it is currently) of the VCR can also be retrieved.

If the owner of a session (the user who started the playback) controls a session with several receivers, all receivers will see the same change in the playback. Currently, we do not have any plans to incorporate functionality for group-switching as described in [12].

Also note that there is no major buffering[4] done in the clients and that e.g. a fast-backward action implies that already viewed data is retransmitted to the client.

Data format

Each recorded data-stream is stored in two files, a *data-file* and an *index-file*. The data-file contains the data- and control-packets in the original session. It also contains the time-stamp of when the packet arrived. The index-file contains byte-offsets into the data-file to make random access of the data faster. The index-file also contains information about the original session.

A recording can also include an optional third file, a *marker-file*. This file contains provider-created markers which can be compared to book-marks in a book. It allows for the viewer of a session to jump to certain points in the session. This could be parts in a movie, a lecture series or it could be markers for each new slide in a presentation.

When designing the mMOD, experiments with data-compression were conducted using simple file compression. The *gzip*-format was used (compression of audio and video data gave about 10% storage-gain, and other media, such as whiteboard, gave up to 90% gain). The VCR can be instructed to use the gzip-compression method but it is default off.

2.2 The Web-controller

The Web-controller is a program that acts as the Web-interface of the mMOD system. Using this interface new sessions can be started, running sessions can be viewed and controlled and a user can join a running session.

Starting a new session

Available sessions are configured easily in a configuration file which the Web-controller parses each time it is started. From this list of sessions an HTML-page containing the available sessions and information about them is automatically created.

The user can be select which session to start and is then presented by an HTML form. This form includes questions about how the session should be played back and where to play it to. A session can be played back using unicast or multicast. The playback destination can be specified in three ways; the user can enter all necessary address-information into

[3]We currently use TCP to get a reliable control-channel and initial tests indicate that the extra delay due to TCP-characteristics are not a problem for the user.

[4]This is usually done in set-top-boxes to make fast-forward and backward easier.

the form, the Web-controller can make a random selection of address-information for the user, or an previously announced session can be selected from a list of known sessions. Information about known sessions is retrieved by the Web-controller from either the mSD (multicast Session Directory) [9] or from the cache of an SDR (MBone Session directory tool) [16].

A new session is started in paused mode by default to give the receivers a chance to start the necessary tools for receiving the session.

Sessions are also by default started with a receiver timeout of 5 minutes, i.e. if no feedback is received within this period the player exits.

The user starting the session can also choose to start only one or more media out of all available media.

Session control

Information about running VCRs can be displayed on user request. The user can choose to view information about all running VCRs or just the VCRs started by that particular user. The information is presented using an HTML-page.

From this HTML-page, a VCR can be controlled using the Web-controller (by links in the HTML-page) or by a Java-applet that connects directly to the VCR.

The Java-applet displays a text-window containing information about the VCR and a set of control-buttons (much like a normal VCR). A shorter response time is achieved, by using an applet to control the VCR instead of using the Web-controller.

Due to the Java security restrictions in Web-browsers, an applet can only make an IP-connection to the host it was downloaded from. This might become a problem if VCRs are running on several machines and the administrator does not want to start a Web-server on each of these machines. In mMOD this is solved by including a very minimal HTTP[5]-server in each VCR. The HTTP-server is integrated into the VCR on the same port as the control-port and the VCR chooses different functionality for the request depending on if it is a HTTP-request connection or a control-connection.

A distributed HTML-presentation system called *mWeb* [10] can be used to control a session if it contains slides presented using mWeb. By selecting a slide in mWeb the VCR is instructed to start playing back from where the slide was first presented.

Joining a session

A user can join a session either by starting the necessary tools by hand, or by selecting the *launch-link*. This launch-link will send back an SDP-file containing all necessary information for joining the session. If the web-browser is correctly configured for handling the MIME-type application/x-sdp the session can be joined using the *mLaunch* program [8].

mLaunch is a program that parses an SDP-file and starts corresponding tools for joining and receiving the session.

2.3 Security issues

When a session is started the user can specify an optional user-name and password that will be used to protect the VCR from unauthorized usage. Each user that wants to control that particular VCR must then identify himself with the correct user-name and password before being able to control it.

If no user-name and/or password is specified the VCR is by default protected from all requests coming from other hosts than the original start request.

These passwords can be overridden by a special administrator password with higher priority. This to allow the administrator to stop and control VCRs.

2.4 Random access in playbacks

A VOD-system usually only includes continuous media such as audio and video. These kinds of media are very easy to play back when random access is involved as the player can just ignore the skipped parts and start playing from the new position in the data.

For other media that are not as continuous as audio and video, i.e. whiteboard data, the situation is much more complex. Here special measures have to be taken for each new medium. In the case of whiteboard data, a *fast forward* means that all skipped data must be played immediately to get a WhiteBoard that is up to date. In the case of *fast backward* the player would want to undo an earlier playback and start playing at an earlier point in the file. This is however not possible unless the player has some semantic knowledge about the data-stream. (If the player had this knowledge, it could parse the data-stream and send undo commands, i.e. delete undoes a draw. to revert to an earlier point in the playback.)

In the current implementation (see section 3) the player skips intermediate parts for media with the names[6] *audio* or *video* and plays intermediate parts for all other media.

Most audio/video MBone applications include a dynamic playback buffer based on the jitter in the arrival of data-packets. If random access is done on IP-level (i.e. not changing the RTP-packets), the jitter-buffer

[5]HTTP is the protocol used on the WWW for retrieving information objects.

[6]The player relies totally on the name and does not try to parse the data-stream to figure out its type.

calculations will become incorrect and the client might decide to just drop the packets or delay longer than needed. To solve this, the session has to be played back on RTP-level and the server has to change the time-stamp in each packet before retransmitting them. Another solution, that works quite well, is to minimize the jitter-buffer in the client[7].

2.5 Server resources

As with any Video/Media-on-demand system, resources are always limited. The mMOD-system limits both the number of total concurrent playbacks and the number of concurrent playbacks requested by a specific host. This unfortunately creates an unbalance between sites where all users access the Web through a proxy-server (for cache- or security-reasons) and sites where each user accesses the Web directly.

3 Implementation and Status

As one of the goals was to create a portable system, the Java programming language [5] was chosen. mMOD is completely written in Java version 1.0.2 (with the necessary fixes for using multicasting [4]) and has been tested to run under both Windows95/NT4 and SUN-Solaris. The only requirement mMOD puts on the WWW-server is that is must support the CGI (Common Gateway Interface) [2] for communication between the Web-server and the mMOD Web-controller.

The current prototype was developed and tested on a SUN workstation running Solaris.

A number of different media and applications are today supported by mMOD, including; any RTP compliant audio/video tool (VIC/VAT/RAT/FPhone), mWeb (a distributed HTML-presentation-system), the MBone WhiteBoard (WB) and the MBone shared text editor (NTE/NetTextEditor).

More information about mMOD and the current implementation is available from [1].

3.1 Further issues

A problem with existing MBone audio/video tools is that there is no inter-media synchronization. This means that if one medium gets delayed the presentation can get unsynchronized.

We would in the future like to include an easily used media-editing tool which would allow content-providers to edit the content of a session.

[7]This is done in the current implementation of mMOD by including a special attribute in the SDP-packet returned on a launch-request. mLaunch understand this attribute and starts vic and vat with the jitter-buffer set to a maximum of 1 second (see section "Joining a session" on 4).

The system has been used for a couple of months now, mainly for storing electronic meetings and lectures in an under-graduate course in distributed multimedia. The initial feedback has been very positive but any real user study is still to be done.

4 Summary and Conclusions

In this paper we have presented a system for recording and playback of MBone sessions including not only audio and video but also several other media available today. Playback can be requested easily through a Web-interface which allows for random access and control of the playback. Available sessions are easily configured by the content-provider.

The system has been used for recording live sessions on the MBone, movies played used the MBone technology as-well as distributed meetings with great success.

The initial feedback from users of the system has been very positive.

Acknowledgments

Thanks to Håkan Lennestål, Johnny Widén and Ulrika Wiss, CDT, for interesting comments and nice encouragement and feedback.

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] *The multicast Media-on-Demand system*, Peter Parnes <URL: http://www.cdt.luth.se/~peppar/progs/mMOD/>

[2] *Common Gateway Interface*, NCSA <URL: http://hoohoo.ncsa.uiuc.edu/cgi/>

[3] *The Interactive Multimedia Jukebox Project*, Kevin C. Almeroth, Mostafa H. Ammar <URL: http://www.cc.gatech.edu/computing/Telecomm/IMJ/>

[4] *Java multicast fixes*, Peter Parnes <URL: http://www.cdt.luth.se/~peppar/java/>

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

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

[7] *The MBone VCR*, Wieland Holfelder, <URL: http://www.icsi.berkeley.edu/mbone-vcr/>

[8] *mLaunch*, Peter Parnes <URL:http://www.cdt.luth.se/˜peppar/progs/mLaunch/>

[9] *multicast Session Directory - mSD*, Peter Parnes <URL: http://www.cdt.luth.se/˜peppar/progs/mSD/>

[10] *mWeb*, Peter Parnes <URL:http://www.cdt.luth.se/˜peppar/progs/mWeb/>

[11] S. E. Deering, *Multicast Routing in a Datagram Internetwork, PhD Thesis*, Stanford University, Dec. 1991.

[12] *On the Use of Multicast Delivery to Provide a Scalable and Interactive Video-on-Demand Service*, Kevin C. Almeroth, Mostafa H. Ammar, Journal on Selected Areas of Communication, August 1996.

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

[14] *RTPTools*, Henning Schulzrinne, <URL: ftp://gaia.cs.umass.edu/pub/hgschulz/rtptools/>

[15] *Session Description Protocol - work in progress*, Mark Handley, Van Jacobson

[16] *Session Directory tool*, Mark Handley, <URL: http://mice.ed.ac.uk/mice/archive/sdr.html>