

AUDIO MEETING HISTORY TOOL: INTERACTIVE GRAPHICAL USER-SUPPORT FOR VIRTUAL AUDIO MEETINGS

David M. Roy

Natural Interactive Systems Laboratory
Odense University
Forskerparken 10
DK-5230 Odense M
Denmark
+45 63 15 73 03
roy@nis.sdu.dk

Saturnino Luz

Natural Interactive Systems Laboratory
Odense University
Forskerparken 10
DK-5230 Odense M
Denmark
+45 63 15 73 11
luzs@acm.org

ABSTRACT

Interactive graphical user-support within an internet-based virtual meeting-place is provided by an Audio Meeting History Tool. The tool addresses the representation, storage, navigation and processing of meeting memory where speech is assumed to be the primary modality of interpersonal communication. Communicative turns are integrated with non-acoustic data to form the meeting history through a GUI component based on a "musical score" metaphor.

1. OVERVIEW

The Audio Meeting History Tool provides interactive graphical user-support within an internet-based virtual meeting place where users may be connected via heterogeneous systems (e.g. workstation, PDA, or mobile telephone) and where spoken language is the main modality of interaction. Features of the Mbone are exploited in a novel way to enable the automatic segmenting and labelling of communicative turns (CTs) to create "structured audio" [1]. CT data are integrated with non-acoustic data to form the meeting history. The tool is accessed through a GUI component based on a "musical score" metaphor which has been designed to increase awareness and support the identification of meeting events using human pattern recognition.

A server-based automatic speech recognition module and meeting activity filter/processing module will be integrated with the graphical-user interface (GUI) component in order to support searching of structured audio data. A demonstrator is under development and will be evaluated using an audio meeting data corpus together with user-trials to illustrate the potential of combining human and automatic recognition of the

This research forms part of the Magic Lounge project, an international collaborative project funded by the I3 ESPRIT long-term research programme of the European Commission (<http://www.dfki.ed/imedia/mlounge>).

patterns of communicative turn taking supplemented by relatively poor speech recognition.

2. GRAPHICAL USER INTERFACE

This tool uses the score/time-line representation with event horizon commonly implemented within video and sound editing interfaces. Waibel et al. previously proposed the "musical score" metaphor for representing the audio recording of meetings together with automatic content extraction [2]. We extend the concept to focus on meeting support, to examine schemes that exploit the availability of data such as turn-taking information, and to integrate audio with non-audio meeting activity data.

The GUI provides a visual representation of the meeting history, which can be viewed at different time-scales (figure 1). Multiple windows can be used to examine different parts of the meeting. Each part or the whole of the window can be played back. Various playback schemes are possible, e.g. clicking on a communicative turn to playback that turn. A number of schemes for fast playback will be considered. The functionality of this tool can be further extended through integration with other meeting helper technology such as an agenda tool, so that the agenda tool provides the information to segment the meeting with hyper-links to the audio meeting history.

Each object type that form the meeting history can be indicated by a corresponding graphic (e.g. 3D coloured bars, representing communicative turns). The display will allow for overlaying text, or maybe the speech signal or other information on the bars.

Initially, the basic display component (see figure 1, top) should operate in two modes: *fixed duration window*, where the communicative turns appear at the event horizon on the right and disappear from the window on the left, and *automatic scaling window*, which constantly scales to fit within a fixed width.

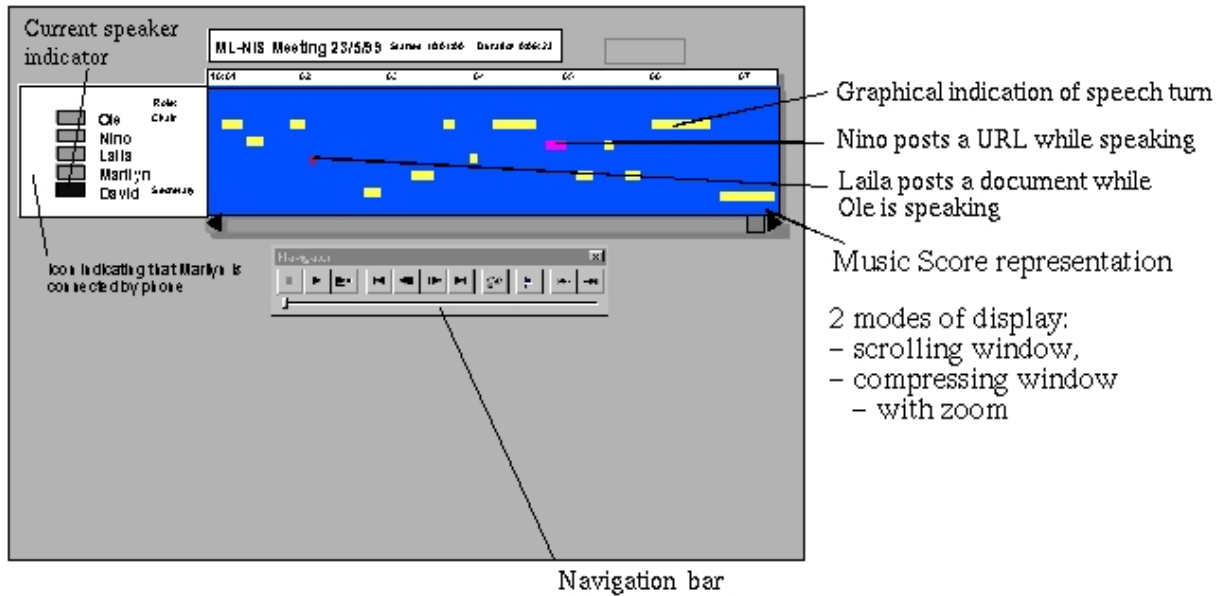


Figure 1: Audio Meeting History Tool GUI

3. SYSTEM ARCHITECTURE

The underlying network structure assumed by the Audio Meeting History Tool is the Multicast Backbone (MBone). The basic client-side tools are adapted from existing MBone audio tools which are based on IETF (*Internet Engineering Task Force*) standards and allow both point-to-point and multicast communication modes. The graphical display described in section 2 (figure 1) is then built on top of one of these tools which keeps track of each user's participation while meeting audio is recorded on a server (figure 2). As previously described, each CT is identifiable as a horizontal bar encoding start-time, duration, and participant's name.

Most of the information needed in order to implement the GUI component is already provided by RTP (Real-Time Protocol) [3], the protocol on which many MBone applications are based. The minimal data structure for the musical-score viewer contains the following items:

1. identification of the meeting,
2. identification of each participant,
3. description of the type of object (payload) carried by each contribution (e.g. streaming audio, chat text, hyperlink, document)
4. start time and
5. end time.

Fields (1) and (2) already form part of the underlying transport protocol (RTP) and respective control data (RTCP). Item (3) is only partially supported in the

core definition of RTP. However, since the protocol allows for the definition of extensions to its profile, future accommodation of non-audio data should be manageable within RTP itself. The GUI interface component should provide for the integration of diverse data transmitted over different channels. Fields (4) and (5) require special post-transmission processing. The data for these fields are extracted from sets of RTP packets and simultaneously stored in the session database at the server side and displayed incrementally at each client.

The proposed system architecture is shown in figure 2. The server side consists of two main components: (a) a daemon based on a standard MBone audio tool which records the audio data exchanged during a meeting and control-data similar to that displayed by the client-side tools and (b) daemons which process the audio data recorded by (a), play it back on demand, etc. The functionality of (b) can be improved to include the ability to query the audio database by topic, speaker, entity etc. This extra functionality builds upon automatic speech recognition techniques and recent advances on topic detection technology [4] and information extraction [5].

In a typical use-case scenario, the MBone audio tool acts as a mixer and does all packet management and audio encoding/decoding necessary to optimise the audio channel. Since the data structure used by the graphical display of the dialogue is extracted directly from RTP data headers, this part of the system does not depend on automatic speaker identification or speech recognition, thus keeping the processing load on client side at acceptable levels. More demanding tasks, such as speech-to-text processing, which might result in extensions and/or changes to the meeting history display

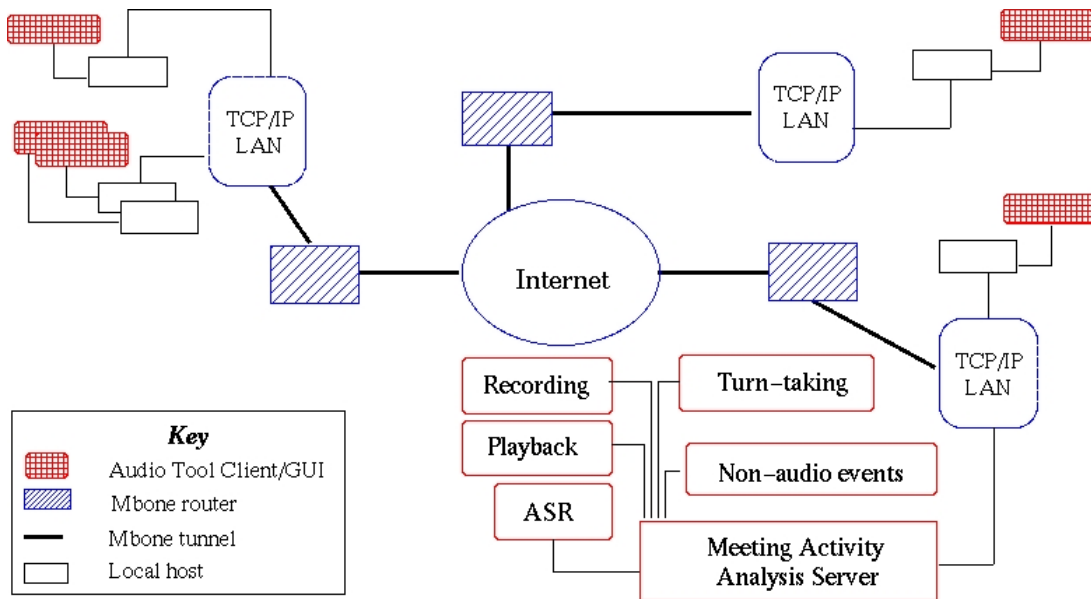


Figure 2: Audio Meeting Tool Architecture

(e.g. by tagging sets of CTs as containing certain keywords, topic etc) may be handled by the server in the background while the meeting is in progress.

3.1. Recording and Playback

For the moment, a simple video recorder type playback will be possible. In addition, it will be possible to click on the CT bar and to playback the associated audio or text etc. Later, it should be possible to manipulate items in the meeting record i.e. apply filters, copy and paste selectively etc.

Two types of use are envisaged for the playback function:

1. the user is looking at data from a meeting that is in progress (and should be able to browse the graphical representation of earlier parts of the meeting), and
2. the user is looking at data from a meeting that has ended.

It should be possible to play a meeting or browse, but it should be assumed that the data is streaming in real time so that either in real time or play back mode there should be an event horizon. Since the meeting data available for playback is stored on a server along with a meta-representation of communicative turns, occasional mismatches might occur between the data displayed on the client's screen at the time when the meeting actually took place and the data subsequently generated by the server. This is due to the fact that it is virtually impossible to guarantee homogeneous data reception at the server and at each client. Multicast protocols allow for substantial packet loss, and even for user-driven specification of a "time-to-live" for each

packet transmitted (which specifies how far into the multicast network a packet will be propagated), yielding scenarios where, for instance, two participants of the same meeting could hear different groups of participants. For the moment, we have chosen to display the data at the client side as encoded in the RTP packets rather than implement yet another (server-based) level of control on top of RTP in order to provide a uniform view of the meeting (i.e. the server's view) for its participants.

4. FURTHER ISSUES

4.1. Delay, poor synchronisation, and interruption of data stream

Transmission of data over the multicast network introduces a variable amount of delay, poor synchronisation and interruption of stream. It is important to determine how best to deal with phenomena and how to represent them on the display.

It is proposed that delay will be accommodated for by allowing the position of the event horizon for each CT to vary in line with the true start-time. Each bar should grow from the event horizon position defined by the start-time of the CT as encoded in the Mbone data (see figure 3). It is necessary to assume that the end time of the CT is not known (as is the case for speech data) until it is finished.

Thus, all participants' windows display the CT at the time that each speaker actually started. When a packet arrives with a visible delay we display it in line with its time stamping – i.e. the event horizon is normally delayed by a varying amount. This should give a subtle indication of the level of delay in the system.

4.2. Synchronisation and pauses

Using the scheme described above, everyone should be working in with the same display. To achieve this, consistent time stamping is important. If the apparent start-time is the real start-time, then each terminal must do the stamping and clocks of all participants must be and remain synchronised.

Another issue is how to deal with and represent pauses where the same speaker resumes speaking. Our initial strategy will be to treat this as a single CT if the pause is below a certain threshold.

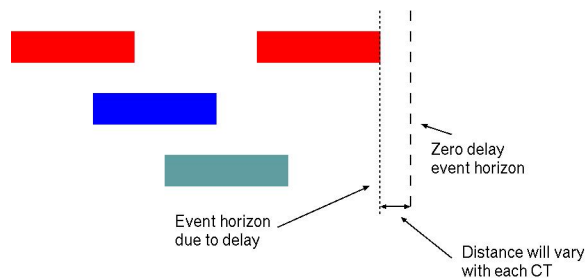


Figure 3: Synchronisation and Event Horizon

5. IMPLEMENTATION

An early prototype of the system is being implemented in Java and C++. Currently, the implementation relies heavily on original C++ code reused from traditional Mbone tools (VAT, in this case) with the GUI being developed in Java. Given the need for interoperability and that the envisaged user tests will be performed using different hardware platforms, we plan to implement a new version of the client tools, totally developed in Java, using features of the new *Java Media Framework* at a later stage.

The Audio Meeting History Tool will provide a platform for integration with various non-audio components of virtual meetings, such as agenda and text posting tools. The initial prototype will allow us to collect and experiment with a corpus of Mbone audio meeting data. This will provide us with a baseline level of functionality against which additional enhance features can be judged.

6. REFERENCES

- [1] D. K. Roy and C. Schmandt, "Newscomm: A hand-held interface for interactive access to structured audio," in *Proceedings of CHI '96*, pp. 173–180, ACM, April 1996.
- [2] A. Waibel, M. Bett, M. Finke, and R. Stiefel-hagen, "Meeting browser: Tracking and summarizing meetings," in *Proceedings of the DARPA*

Broadcast News Transcription and Understanding Workshop, (Lansdowne Conference Resort Lansdowne, Virginia, USA), Feb. 1998. <http://www.nist.gov/speech/proc/darpa98>.

- [3] H. Schulzrine, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," tech. rep., IETF, February 1999.
- [4] C. L. Wayne, "Topic detection and tracking (tdt): Overview and perspectives," in *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, (Lansdowne Conference Resort Lansdowne, Virginia, USA), Feb. 1998. <http://www.nist.gov/speech/proc/darpa98>.
- [5] R. Grishman, "Information extraction: Techniques and challenges," *Lecture Notes in Computer Science*, vol. 1299, 1997.