Ubiquitous Audio: Capturing Spontaneous Collaboration

Debby Hindus
Chris Schmandt

MIT Media Lab
20 Ames Street
Cambridge, MA, USA 02139
Email: hindus@media.mit.edu or geek@media.mit.edu

ABSTRACT
Although talking is an integral part of collaborative activity, there has been little computer support for acquiring and accessing the contents of conversations. Our approach has focused on *ubiquitous audio*, or the unobtrusive capture of voice interactions in everyday work environments. Because the words themselves are not available for organizing the captured interactions, structure is derived from acoustical information inherent in the stored voice and augmented by user interaction during or after capture. This paper describes applications for capturing and structuring audio from office discussions and telephone calls, and mechanisms for later retrieval of these stored interactions.

KEYWORDS
Stored voice, semi-structured data, ubiquitous computing, collaborative work, software telephony, multimedia workstation software.

INTRODUCTION
People spend much of their workday talking. In Reder and Schwab's recent study of professionals, phone calls comprised about 20% of the workday, and face-to-face meetings accounted for an additional 25-50% [23]. Yet this time spent talking has been to a great extent out of reach of computer technology. This loss of speech information is all the more striking, given the dominance of the audio medium in influencing communication outcomes regardless of the presence of visual and other media [22]. Furthermore, speech communication fulfills different communicative purposes than text communication and "is especially valuable for the more complex, controversial, and social aspects of a collaborative task" [4].

Nevertheless, audio is, in a sense, a forgotten technology for CSCW. Multimedia has received considerable attention as a CSCW technology area in recent years, and many applications have focused on synchronous video communication. Because the social and technological issues involved in supporting video calls are so demanding, synchronous video conferencing remains a research topic [7, 12, 20], whereas audio communication is already part of daily life—telephones sit on every desk. Eventually, however, simply supporting synchronous conversations will be insufficient and the research emphasis then will shift to storage and retrieval of the *contents* of interactions.

In many ways, storing and retrieving communication for later review is much more demanding than merely establishing the synchronous channel. Both audio and video are time-dependent media with few sharp boundaries for indexing or classification. If it were practicable, speech recognition and natural language processing techniques could convert speech to text. Such processing will not be reliable and as fast as human speech for perhaps decades [35], and in any case would cause the loss of nuances carried by the audio signal but not in the words themselves. In the meantime, extending audio technology to spontaneous conversation will require automatic derivation of structure without understanding the spoken words. This *semi-structured audio* will assist in later retrieval as well.

This paper is about various means of capturing voice interactions in everyday work environments; we call this *ubiquitous audio*. We discuss applications for capturing audio in work situations and deriving structure during or after capture, leading to semi-structured audio. We describe user interfaces for later retrieval of these stored voice interactions. Finally, we briefly mention how voice records of meetings and conversations fit into the broader context of utilizing voice as a datatype.

BACKGROUND
To motivate ubiquitous audio, we will first review the current role audio plays in CSCW. Audio as a communication medium has received attention in computer conferencing applications, although the related technological problems have obscured its potential as a source of data. Recorded speech has been used in a limited role, in applications that require little structuring of the audio data. Structure is what is needed

to make audio more useful in CSCW, and we describe semi-structured audio and its antecedents. Looking ahead to ubiquitous computing, stored voice will be more readily available, and access and retrieval will be significant issues. A glimpse of this future is available by examining current projects.

Audio in CSCW Applications

Audio in most CSCW applications is only a medium for synchronous communication and not yet a source of data. Audio conferencing over telephone lines has proven to be of limited utility. In teleconferencing applications, echo cancellation and feedback are common problems, particularly with speakerphones that are half-duplex (i.e., only one party can speak at a time) [8]. This has led to computer conferencing systems that support voice through a custom network, such as MMConf [11]. A recent desktop conferencing system, MERMAID [30], uses ISDN and so can support full-duplex conversations; that is, all parties can talk simultaneously.

Short pieces of recorded speech—voice snippets—are the main use of speech as data in current CSCW applications. These snippets are used in message systems, e.g., voice mail, and in multiuser editing systems. (The categories follow Ellis, et al.'s taxonomy [8].) Voice can also be used to annotate text, as was done in Quilt [9]. Voice snippets can be stored in a personal calendar or within spreadsheet cells. The stored voice is not itself structured in these applications; the recorded speech is treated as a single unbroken entity and the application maintains an external reference to the sound, such as a message number, calendar date, or position within the text or spreadsheet.

Semi-structured audio

Semi-structured media have not, to date, been extensively explored. In the text domain, Information Lens is an example of the power of a semi-structured approach [18]. Information Lens users add extra fields to electronic mail messages; this extra information provides attributes to messages but is not part of the message body itself. Users can then write rules to route and sort received mail, based on these additional attributes [17].

Similarly, semi-structured audio provides a framework for organizing and manipulating audio data. Information inherent in the speech signal and in the audio interaction can be combined with situational constraints and user annotations to create semi-structured audio data. An early example of semi-structured audio was Phone Slave [27]; it used conversational techniques to take a telephone message, asking callers a series of questions and recording the answers. These sound segments could be highly correlated with structured information about the call. For example, the response to "At what number can you be reached" contained the phone number. Recently, structured data capture has been applied by Resnick [24] to a voice-driven application that uses the form-entry metaphor for a telephone-based voice bulletin board. After recording their messages, contributors to the bulletin board are asked to fill in additional fields, using appropriate mechanisms. For instance, the headline field is filled in with a brief recording and expiration dates are given by touch tones.

Ubiquitous computing

Technological advances are leading to small, portable computers that will communicate through wireless networks; current examples are laptops, palmtops, alphanumeric pagers, and smart cellular phones. Xerox PARC is exploring technologies ranging from electronic whiteboards to wearable computers in the context of *ubiquitous computing*: the concept that explicit computer interactions will be replaced by specialized smart devices that are unobtrusively present in all aspects of day-to-day pursuits [31].

The advent of ubiquitous computing will permit the acquisition of previously inaccessible data about daily activities. The Activity Information Retrieval (AIR) project at Rank Xerox EuroPARC has been exploring how ubiquitous computing can provide people with access to their own previous activities. In one effort, laboratory members wear "active badges" (originally developed by Olivetti) that can be tracked [29], and personalized diaries are created that summarize each badge wearer's whereabouts and meetings. Another component of AIR is a system that continually videotapes lab members. The stored video can be retrieved by using situational information such as where the person was at a particular time, and by using timestamped notations made on pen-based devices during meetings [16].

A portable prototype for capturing user-structured audio has been developed at Apple. A handheld tape recorder was modified so that users could mark interesting portions of recordings of meetings or demarcate items in personal memos. A key point is that these annotations could be made in real time, as the sound was being recorded [5].

THE HOLY GRAIL: AUTOMATIC TRANSCRIPTION OF FORMAL MEETINGS

Conspicuously absent from our discussion so far is the notion of capturing the spoken contents of formal group meetings without human transcription. Given the importance of meetings, this is an obvious CSCW application, as indicated by the body of work on electronic meeting systems [19, 6]. Due to technological issues, however, it is very difficult to automatically structure recordings of meetings.

The first issue is obtaining high-quality recording of the participants' speech. Individual microphone placement is critical and may encumber the participants. Using a single wide-area microphone is simpler, but will compromise the audio quality.

The next issue is associating each utterance with a participant. Ideally, each person's speech would be recorded on a separate audio channel. But room acoustics make it quite difficult to get each attendee's speech to be transmitted by only one microphone, and using highly directional table microphones greatly constrains participants' movements. For large groups, an auto-directional microphone can orient itself towards the podium or towards the audience [10], although its discrimination when talkers are sitting side-by-side is unproven. Signal processing techniques based on spectral analysis might be capable of determining who is talking from among multiple talkers on one recording; these techniques

are being researched.

Transcription of the spoken words is the third issue. Speech recognition of fluent, unconstrained natural language is nowhere near reality yet. Even keyword spotting, to produce partial transcripts, is very difficult when applied to spontaneous speech, especially from multiple talkers [28]. However, wordspotting techniques have recently been incorporated into an audio indexing and editing system [32], and keyword spotting need not be perfect; the Intelligent Ear [26] was an early system that made use of a graphical color display and indicated the confidence of the word recognition through luminance levels.

THE CAPTURE AND STRUCTURING OF SPONTANEOUS AUDIO

Common workaday activities other than formal meetings provide a starting point for exploring ubiquitous audio, both in terms of user interfaces and audio processing. Ubiquitous audio can come from a number of sources and through a variety of physical input devices. The ubiquitous computing approach will eventually lead to large quantities of stored information; for example, all the conversations that occur in an office in a year could be saved on a single disk drive *.

The semi-structured approach defines a framework for making these quantities of audio usable by incorporating acoustical cues, situational and contextual data, and user-supplied structure. Acoustical structure includes speech and silence detection, and associating portions of the audio signal with the correct talker. Semi-structure provides enough structure for flexible access to the data later, without relying on the explicit creation of structure by the source or recipient of the audio. Users *can* create structure as they see fit, but they do not *have* to create structure in order for the audio data to be manageable and accessible.

We have explored ubiquitous, semi-structured audio in varied tools that support informal meetings, personal notes, and telephone conversations. This section describes two such tools and issues in structuring the audio they capture. The first of these applications is a digital "tape loop" that provides short-term auditory memory in the office, with no inherent structure to the recording. The second application allows users to selectively mark and save portions of telephone conversations, and utilizes the conversants' turn-taking for structure. For each application, we consider both the capture phase, during which the spoken collaboration is recorded and accessed immediately thereafter, and the retrieval phase, wherein the stored voice can be used at a later time.

Capturing office discussions

When multiple authors are working on a collaborative writing project, one may suggest a new wording to a paragraph, which the other strives to write down but cannot. By the time the second author says, "That was perfect, say it again," the words have already been forgotten. `Xcapture` is meant to supply exactly this short-term memory; the user remembers the flow of the recent conversation, so `xcapture` makes no direct use of any structure in the recorded audio.

*Assuming eight-hour days, wherein conversing takes four hours (of which 30% is silence), and 10:1 compression of telephone-quality speech, a year's worth of office speech would require approximately 2 gigabytes of storage.

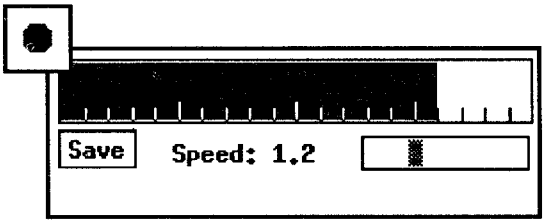


Figure 1: `Xcapture` after an office discussion.

Many workstations are now equipped with a speaker and microphone, but in practice the microphone is rarely used. `Xcapture` provides short-term audio memory in the office by making use of workstation resources in a background task. Whenever the microphone is not in use by another application, `xcapture` records ambient sound into a circular buffer; five to fifteen minutes is a typical length.

Retrieving office discussions

`Xcapture` records until the user clicks on its animated icon, causing a new window to appear. This window displays the entire audio buffer using a SoundViewer widget (see Fig. 1). The SoundViewer, used extensively in our work, provides a direct manipulation user interface to sound files. Time is displayed horizontally as tick marks. When the user clicks on the SoundViewer, the sound plays and a cursor bar flows into the window from left to right. The mouse can be used to move this cursor and cause the replay to jump to the new location; that is, the mouse provides a time-based offset into the sound, allowing random access.

During retrieval, the `xcapture` user is faced with the task of finding the interesting speech segment from within multiple minutes of recorded audio. The SoundViewer indicates time, but gives no clue to the contents of the sound it represents. A longer recording buffer makes `xcapture` more useful, but retrieval is onerous for a lengthy recording, even with the random access mechanism described above. Therefore, `xcapture` and the SoundViewer support scanning through the recorded audio by replaying the speech back in less time than was required to record it. A slider under the sound viewer allows the `xcapture` user to increase playback to up to three times normal speed; although intelligibility is significantly reduced beyond twice normal speed, the upper limit allows faster scanning through familiar material.

Time-compressing speech so that it remains intelligible is not straightforward. Simply increasing the rate at which samples are played is inadequate; this raises the pitch, as with the cartoon chipmunks. Discarding chunks of sound during playback is a better approach; quality is improved by finding and discarding complete pitch periods, and smoothing the boundaries. For a survey of time compression techniques, see Arons [2].

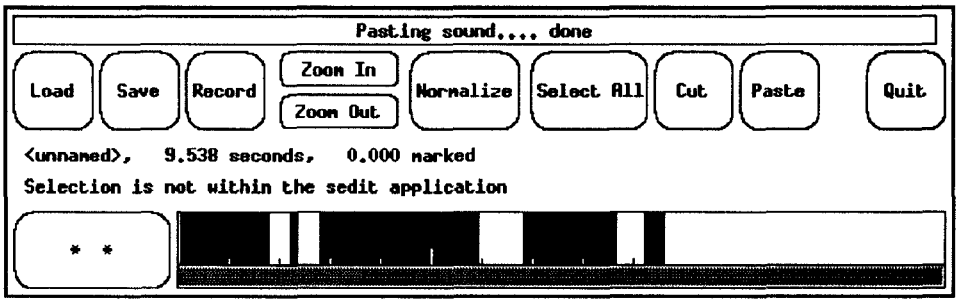


Figure 2: A sound editor showing speech and silence intervals

Xcapture allows the entire buffer to be saved to a file, but this does not turn out to be very useful. Instead, using the mouse, the xcapture buffer can be cut and pasted into an audio editor that displays speech and silence intervals (see Fig. 2), and from there a snippet of a few seconds can be pasted into other audio-capable applications such as a calendar (see Fig. 3) or things-to-do list.

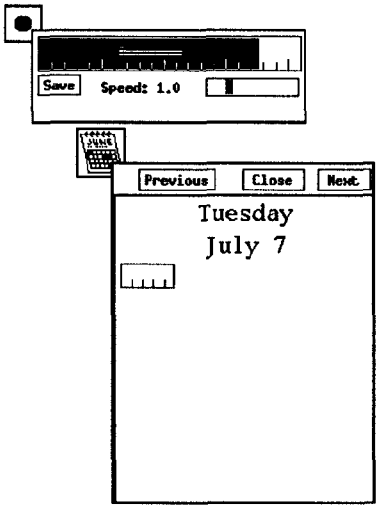


Figure 3: Sound fragment being pasted into a calendar entry

Xcapture provides a means of transforming ubiquitous speech into manageable chunks that can be incorporated into databases that supply structure, e.g., the short list of things to do on a particular day. But the lack of structure limits the utility of xcapture; even for short-term memory, audio structure is required for direct manipulation interfaces. We provide an improved interface incorporating structure in an application to record telephone calls.

DYNAMIC CAPTURE AND DISPLAY OF TELEPHONE CONVERSATIONS

Xcapture supported structuring during retrieval; this portion of our work addresses creation of structure at the time of capture and allows interaction during capture. In addition to structuring during capture, this project addresses the visual representation of free-form conversations and dynamic displays for real-time capture. It features the Listener, a telephone listening tool that allows users to identify and save the relevant portions of telephone calls as the phone conversation progresses. Telephone conversations are a practical choice for demonstrating semi-structured audio; very little equipment is required beyond audio-capable workstations, and who is speaking can be detected because the two audio channels can be separated. In this sense, phone calls can be thought of as two-person meetings. Also, telephone conversations will be increasingly mediated by workstations, so situational information can be associated with a digitized telephone conversation [34, 15].

The Listener captures structure from telephone calls, as described in the following scenario: You receive a telephone call. The Listener pops up a notification window on your screen. You choose to record the call. While you are talking, a graphical representation of the conversation is constructed on your screen, showing the shifts in who is speaking and the relative length of each turn. You can click on a segment to indicate that it should be saved. At the end of the phone call, you can listen to segments and decide which ones to save, or just save all the marked segments.

Capturing and Displaying Conversational Structure

Studies of audio communication provide parameters for automatically structuring conversations. Business telephone calls are brief and therefore tractable; the expected duration of business telephone calls is 3–6 minutes [23]. Speakers alternate turns frequently, and utterances are typically either very short, or 5–10 seconds long [25, 33]. Furthermore, turn-taking pauses are not distinguishable from other pauses by their length, and many turns happen with minimal pausing [3]. The inherent structure of a conversation therefore consists of the naturally-occurring pauses at the end of phrases and sentences and the alternation of utterances between talkers. Audio data can be automatically segmented into understandable pieces on the boundaries between talkers and between phrases.

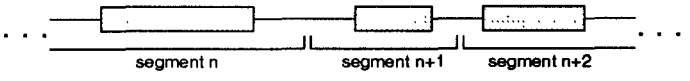
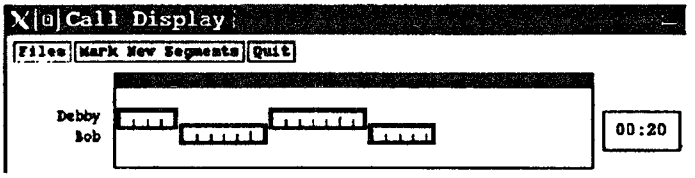
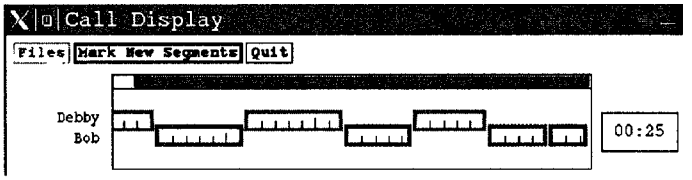


Figure 4: Silences are divided between segments

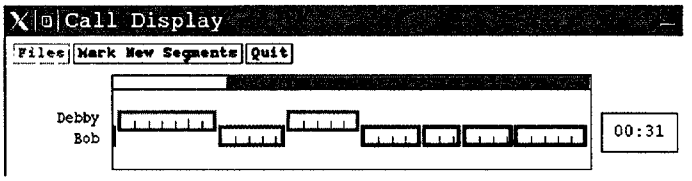
Two microphones collect audio signals for the Listener. One



D: Hello, this is Debby Hindus speaking.
B: Hi Deb, it's Bob. I'm just getting out of work, I figured I'd call and see how late you're going to stay tonight.
D: Well, I think it'll take me about another hour, hour and a half, to finish up the things I'm doing now.
B: OK, I'm just going to head on home, I'll probably do a little shopping on the way.



D: Well, if you think of it, maybe you could get some of that good ice cream that you got last week.
B: OK. By the way, somebody, uh...
B: mentioned an article you might be able to use



B: in your tutorial. Debby: Oh really? [Debby's very short turn is ignored.]
B: Yeah, it's by Graeme Hirst, in the June '91 Computational Linguistics.

Figure 5: Sequence of segments during a phone call, with transcriptions

is connected to the telephone handset and carries speech from both talkers. The other microphone sits in the user's office near the telephone and carries just that person's speech (assuming that the handset is used rather than a speakerphone). This second, single-talker audio stream enables the Listener to distinguish between the two talkers. The Listener receives audio data from both mikes, performs pause detection on each source, synchronizes the sources, and then locates changes of talker between pauses. This last step is needed because turn-taking pauses can be undetectable with just pause detection. To determine the segment boundaries, the Listener incorporates a state machine that takes into account who is talking, the previous state, and the segment duration. As shown in Fig. 4, segment boundaries are calculated so that they fall in the pauses between utterances; this makes segments sound complete when played individually. The new segment is then added to the call display. (Other visual representations have shown speech as periods of sound and silence [1]. Both forms highlight the temporal aspect of sound and provide structure for display and graphical interaction.)

The call display that appears during the conversation must be unobtrusive, so as not to interfere with the conversation. Also, short-term memory constraints imply that only the recent portions of the conversation are salient, and interesting segments can be identified only shortly after the segment takes place.

Figure 5 shows the call display during part of the conversation. As the conversation proceeds, a visual representation of approximately the previous 30 seconds is displayed retrospectively. Each conversational turn is shown, reflecting the phrase-level utterances of the talkers. Each segment, or portion of the audio signal, is displayed within a SoundViewer, the same audio representation that is used throughout our applications. Each tick mark within each SoundViewer represents one second of audio. New segments appear at the right-hand side, and older segments scroll out of view to the left. Segments from each talker can be visually distinguished by their relative positioning and by different border colors when unmarked.

The Listener is an X Window System client application that communicates with several server processes, and it is built on

top of a generic sound-and-text handling mechanism called the ChatViewer [13]. The ChatViewer is a widget designed to manage and display *chats*: collections of recorded speech segments and short text strings.

Adding User-Supplied Structure

The derived conversational structure makes it possible to identify the segments worth saving for later reference. The Listener provides mechanisms for marking, or indicating to the Listener, these interesting segments. When the conversation turns to substantial matters, the user can toggle automatic marking so that all new segments are marked. Furthermore, a user can mark individual segments at any time by clicking on them, and marking is reversible. Marking a segment also highlights it visually.

During the phone call, the user's attention is focused on the conversation and not on interacting with the Listener program. Therefore, the only feasible user actions are clicking the pointer on segments of interest or just toggling the Mark New Segments button. The nature of the user interaction changes from capture to review, however, once the conversation is completed. The entire conversation is available for review at this point through a browsing application. The post-call browser displays the conversation and provides additional editing functions. A user can replay all or part of the conversation, revise the choice of segments to store, annotate with text, and save the chat for later retrieval. Once these post-call revisions are made, only the marked segments are saved.

Browsing Stored Telephone Conversations

Chats may be retrieved long after the phone call took place. There are two levels of retrieval. The first is finding the specific audio segments within a chat that contain the desired conversation extract. The second is locating a particular chat from among numerous stored chats. Our focus has been narrowly focused on capturing and retrieving single conversations. Future efforts will need to address mechanisms for navigating among many chats, making use of the situational data to locate one chat among many in a fashion akin to locating a specific electronic mail message.

Audio cannot be searched in the fashion that text can, and so situational and supplemental structure will be needed to provide memory cues as to the content of the stored audio. The user's choice of which segments to save is one form of supplemental structure; textual tags are another.

The Listener collects and stores contextual, or situational, structure, along with the stored audio. For telephone calls, contextual data includes the other party's name and phone number, if known; the time and date of the call; and the extension of the user. Marked segments will typically occur in consecutive groups, or paragraphs. When the chat is retrieved in the future, these paragraphs are reflected in the chat's layout, as shown in Fig. 6. Users can provide an optional descriptive tag for each conversation and one for each paragraph. Tags are not required; the timestamp is sufficient identification for retrieving a conversation.

DISCUSSION

The goal of this paper has been to introduce the concept of ubiquitous computer gathering of audio and to present applications and user interfaces to manage ubiquitous audio. We do not claim that the applications presented herein have solved the problems associated with managing large amount of semi-structured stored voice. We have presented preliminary work, and we expect to continue this research and extend it into new domains over the next several years.

We have used the applications ourselves enough to be confident that they have value. We have, for example, used the Listener while collaborating long-distance on this paper and mailed impromptu office discussions to group members not present at the time. Our research group has gained significant experience with audio snippets in a number of applications described in this paper, and one of the lessons learned has been the utility of audio cut and paste between applications. The SoundViewer supports a built-in X Windows-based selection mechanism, and so a portion of any sound in any application, including the telephone conversation browser, can be selected and moved to another application.

Obtaining high-quality audio from microphones in offices is still problematic, and acoustic structuring is limited to speech and silence detection. The audio quality from telephones is good, but using two microphones to segment based on talker is awkward and imperfect. When workstations are equipped with ISDN telephone lines, talker-based segmentation will be more realistic because the audio signal from each talker will be carried separately.

We have only begun to explore capturing and structuring large amounts of audio. One future direction, of course, is to start to extend our applications to meetings of more than two persons. Another direction is in new approaches to deriving structure. For example, although real-time speech understanding systems are a number of years away, non-real-time partial transcriptions could be made available as an overnight service by a specialized server in a network. Another approach to structuring is to classify (and perhaps delineate) segments by correlating the prosody of the segments with templates of classes of utterances. For instance, questions can be identified by the ending rising pitch [14].

A final point concerns privacy issues. Office discussions and telephone calls are already vulnerable to inappropriate or surreptitious recording, and conventions and laws govern what is considered appropriate with respect to current recording technology. As the technologies described in this paper become more widespread, users will continue to invent or adapt social protocols for negotiating levels of privacy that suit their needs.

CONCLUSION

The work presented in this paper demonstrates that every-

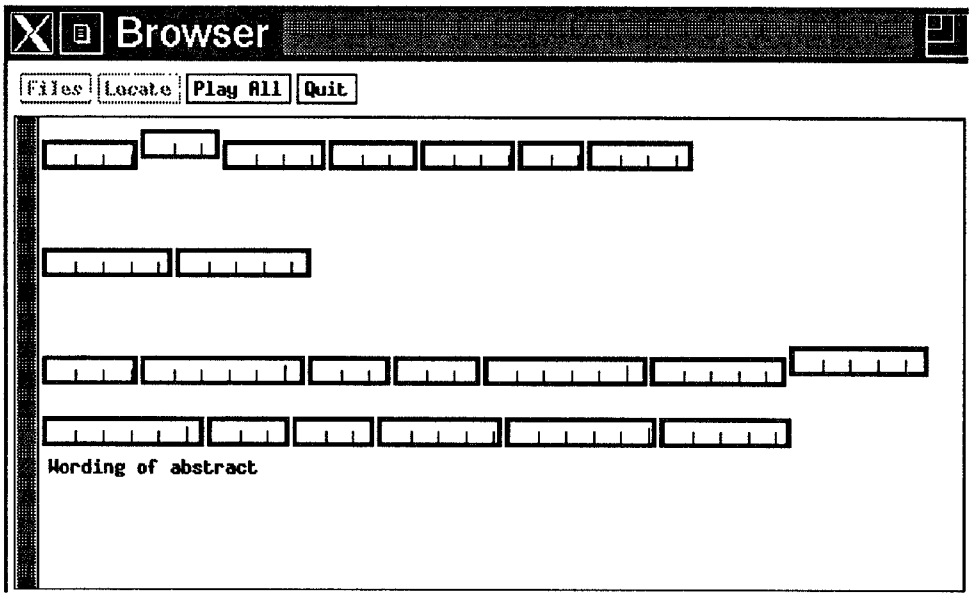


Figure 6: Browsing a stored conversation with three groups of saved segments

day audio interactions can be captured in various ways, and semi-structuring the audio data aids in storage and retrieval. Real-time structuring and display is feasible by making use of situational constraints and by adopting simple segmentation strategies. It is exactly the mundane nature of audio that makes it a worthwhile focus for CSCW. As Moran and Anderson [21] point out, technologies should be mundane in the workaday world and are successful when ordinary activities interact seamlessly with technology. Conversations and telephone calls are examples of such ordinary activities, and they provide a good starting point for exploiting the potential of ubiquitous audio.

ACKNOWLEDGMENTS

The work presented in this paper relies upon software developed by members of the Media Lab Speech Research Group. Barry Arons wrote the audio server used by both `xcapture` and the Listener. The SoundViewer widget was written by Mark Ackerman and Chris Horner, and the mechanics for audio cut and paste were implemented by Sheldon Pacotti. `Xcapture` was written by Lorin Jurow, and `sedit` was enhanced by Jordan Slott. Finally, Lisa Stifelman and Eric Ly worked on related audio applications.

We would like to thank Mark Ackerman for his thoughtful review of an early draft of this paper and Wendy Mackay for her incisive comments on a later draft. She and Kate Ehrlich also contributed to the research on semi-structured telephone conversations. The major portion of this work was funded by Sun Microsystems, Inc.

REFERENCES

1. S. Ades and D. C. Swinehart. Voice annotation and editing in a workstation environment. Technical Report CSL-86-3, Xerox Palo Alto Research Center, Sept 1986.

2. B. Arons. Techniques and applications of time-compressed speech. To appear in Proceedings of 1992 American Voice I/O Society Conference.

3. G. W. Beattie and P. J. Barnard. The temporal structure of natural telephone conversations (directory enquiry calls). *Linguistics*, 17:213–229, 1979.

4. B. L. Chalfonte, R. S. Fish, and R. E. Kraut. Expressive richness: A comparison of speech and text as media for revision. In *Proceedings of the Conference on Computer Human Interaction*, pages 21–26. ACM, Apr 1991.

5. L. Degen, R. Mander, and G. Salomon. Working with audio: Integrating personal tape recorders and desktop computers. In *Human Factors in Computer Systems – CHI'92 Conference Proceedings*, pages 413–418, May 1992.

6. A. R. Dennis, J. F. George, L. M. Jessup, J. F. Nunamaker, Jr., and D. R. Vogel. Information technology to support electronic meetings. *MIS Quarterly*, 12(4):591–624, 1988.

7. C. Egido. Teleconferencing as a technology to support cooperative work: its possibilities and limitations. In J. Galegher, R. E. Kraut, and C. Egido, editors, *Intellectual Teamwork: Social and Technological Foundations of Cooperative Work*, chapter 13. Lawrence Erlbaum, 1990.

8. C. A. Ellis, S. J. Gibbs, and G. L. Rein. Groupware: Some issues and experiences. *Communications of the ACM*, 34:38–58, January 1991.

9. R. S. Fish, R. E. Kraut, and M. D. Leland. Quilt: a collaborative tool for cooperative writing. In *Conference*

on office information systems, pages 30–37, Palo Alto, CA, 1988. ACM.

10. J. L. Flanagan, D. A. Berkeley, G. W. Elko, J. E. West, and M. M. Sondhi. Autodirective microphone systems. *Acustica*, 73(2):58–71, February 1991.

11. H. C. Forsdick. Explorations into real-time multimedia conferencing. In *Proceedings Second International Symposium on Computer Message Systems, IFIP Technical Committee on Data Communications*, Washington, D.C., September 1985.

12. C. Heath and P. Luff. Disembodied conduct: communication through video in a multi-media office environment. In *Human Factors in Computer Systems – CHI'91 Conference Proceedings*, pages 99–103, 1991.

13. D. Hindus. Semi-structured capture and display of telephone conversations. Master's thesis, MIT, Feb 1992.

14. J. Hirschberg and J. Pierrehumbert. The intonational structuring of discourse. In *Proceedings of the Association for Computational Linguistics*, pages 136–144, July 1986.

15. R. Kamel, K. Emami, and R. Eckert. PX: supporting voice in workstations. *IEEE Computer*, 23(8):73–80, Aug 1990.

16. M. Lamming and W. Newman. Activity-based information retrieval: Technology in support of human memory. Technical Report 92-002, Rank Xerox EuroPARC, Jan 1992.

17. W. E. Mackay, T. W. Malone, K. Crowston, R. Rao, D. Rosenblitt, and S. K. Card. How do experienced information lens users use rules? In *Human Factors in Computing Systems, CHI 89 Proceedings*, pages 211–216. ACM, 1989.

18. T. W. Malone, K. R. Grant, K.-Y. Lai, R. Rao, and D. Rosenblitt. Semi-structured messages are surprisingly useful for computer-supported coordination. *ACM Transactions on Office Information Systems*, 5(2):115–131, 1987.

19. M. Mantei. Capturing the capture lab concepts: A case study in the design of computer supported meeting environments. In *Computer Supported Cooperative Work – CSCW'88 Conference Proceedings*, pages 257–270, 1988.

20. S. Minneman and S. Bly. Managing a trois: a study of a multi-user drawing tool in distributed design work. In *Human Factors in Computer Systems – CHI'91 Conference Proceedings*, pages 217–223, 1991.

21. T. P. Moran and R. J. Anderson. The workaday world as a paradigm for CSCW design. In *Computer Supported Cooperative Work – CSCW'90 Conference Proceedings*, pages 381–393, 1990.

22. R. B. Oschman and A. Chapanis. The effects of ten communication modes on the behavior of teams during co-operative problem solving. *International Journal of Man/Machine Systems*, 6:579–619, 1974.

23. S. Reder and R. G. Schwab. The temporal structure of cooperative activity. In *Computer Supported Cooperative Work – CSCW'90 Conference Proceedings*, pages 303–316, 1990.

24. P. Resnick. HyperVoice: A phone-based CSCW platform. To appear in CSCW'92 Conference Proceedings.

25. D. R. Rutter. *Communicating by Telephone*. Pergamon Press, 1987.

26. C. Schmandt. The Intelligent Ear: A graphical interface to digital audio. In *Proceedings IEEE Conference on Cybernetics and Society*, pages 393–397, October 1981.

27. C. Schmandt and B. Arons. Phone Slave: A graphical telecommunications interface. *Proceedings of the Society for Information Display*, 26(1):79–82, 1985.

28. M. Soclof and V. Zue. Collection and analysis of spontaneous and read corpora for spoken language system development. In *Proceedings of ICSLP*, pages 1105–1108, Nov 1990.

29. R. Want and A. Hopper. Active badges and personal interactive computing objects. *IEEE Transactions on Consumer Electronics*, 38(1):10–20, Feb 1992.

30. K. Watabe, S. Sakata, K. Maeno, H. Fukuoka, and T. Ohmori. Distributed desktop conferencing system with multiuser multimedia interface. *IEEE Journal on Selected Areas in Communications*, 9(4):531–539, 1991.

31. M. Weiser. The computer for the 21st century. *Scientific American*, 265:66–75, September 1991.

32. L. Wilcox and M. Bush. HMM-based wordspotting for voice editing and indexing. In *Proceedings of Eurospeech 91*, pages 25–28, Sep 1991.

33. C. Wilson and E. Williams. Watergate words: A naturalistic study of media and communications. *Communications Research*, 4(2):169–178, 1977.

34. P. Zellweger, D. Terry, and D. Swinehart. An overview of the etherphone system and its applications. In *Proceedings of the 2nd IEEE Conference on Computer Workstations*, pages 160–168, Santa Clara, CA, March 1988.

35. V. W. Zue. From signals to symbols to meaning: On machine understanding of spoken language. In *Proceedings of the 12th International Congress of Phonetic Sciences*, August 1991.