Advances in Human—Computer Interaction

Volume 4

Edited by

H. REX HARTSON
DEBORAH HIX

Virginia Polytechnic Institute and State University

ABLEX PUBLISHING CORPORATION
NORWOOD, NEW JERSEY

Printed in the United States of America

ISBN: 0-89391-934-9

ISSN 0748-8602

Ablex Publishing Corporation
355 Chestnut Street
Norwood, New Jersey 07468

CHAPTER 8

From Desktop Audio to Mobile Access: Opportunities for Voice in Computing*

Christopher Schmandt

Media Laboratory
Massachusetts Institute of Technology

With increasingly fast general-purpose microprocessors, it has become feasible to support voice processing on every personal computer or workstation. Digitization and playback are already commonplace; shortly, every computer will include a speaker and microphone. More sophisticated tasks, such as text-to-speech synthesis and voice recognition, will soon be possible on the desktop. Miniaturization of electronic components in the form of the laptop computer is already allowing an increased amount of work to be performed away from the desk. Speech technology already lets travelers keep in touch with the office while on the move through voice mail.

This chapter probes how existing and new software applications can take advantage of voice, and how they will afford entirely new styles of accessing computers. Because voice-processing technology has been, until recently, both specialized and expensive, it has been employed in a limited number of specialized application areas. But with its increasing availability in the workstation, speech has recently acquired the *potential* to become much more broadly used in our daily computer-based work environments. Yet cost reductions and ubiquitous availability by themselves are no guarantee of user acceptance of the tech-

* Several reviewers offered many valuable suggestions on this chapter, including Gayle Sherman, two anonymous reviewers, and especially Barry Arons. In addition to the author, a number of students helped develop the applications discussed in this chapter. This list includes, but is not limited to: Mark Ackerman, Barry Arons, Derek Atkins, Debby Hindus, Angie Hinrichs, Chris Horner, Eric Ly, Sanjay Manandhar, Sheldon Pacotti, and Lisa Stifelman. Support for these projects was provided by Sun Microsystems, Apple Computer, and AT&T.

nology. The success of voice on the desktop and its remote or portable extensions depends on careful selection of appropriate applications and great care to user interaction techniques in order to overcome limitations in the voice channel. The challenge is to integrate voice tightly into our work environments so that, in retrospect, we will wonder how we ever used a mute computer.

The driving force behind widespread acceptance of this new medium of computer interaction will be the possibility of using speech in circumstances for which keyboards and displays are impractical. The use of voice for remote access does not, however, free the workstation from supporting it through conventional screen-based interfaces; the key will be to use voice as a means of merging and integrating different modes of access. Equally important are the richness and expressiveness of speech, as well as the fact that we are continually surrounded by it in our work. The goal of much of the work described in this chapter is to get computers more involved in the modes of communication we use with each other to allow them to more ably assist us on our own terms.

It is all too easy to be overly optimistic and predict voice as part of every desktop application. Unfortunately, speech is a difficult medium to employ successfully, with very demanding user interface requirements. Some of the limitations of speech are inherent in the medium, while others are constraints imposed by current voice-processing technologies. Only by appreciating both the strengths and limitations of speech in the computer interface can we begin to suggest how voice processing may proliferate, and what interaction techniques will enable its widespread acceptance.

This chapter begins with an overview of speech, examining its potential as a powerful tool of human communication as well as its limitations as an interactive medium. It then offers a brief review of the various speech technologies and their specific limitations. Finally it will explore how voice can be employed under three user interaction environments: at the desktop, remote from the workstation over a telephone, and in a highly portable hand-held computer. Examples of work from the author's group at the M.I.T. Media Laboratory will be used to illustrate possible implementations of each style of interaction. But the message of this chapter is less about details of any one of these projects than it is to raise the possibility of incorporating all these ways of using voice into future computer applications.

CHARACTERISTICS OF SPEECH

Speech is invaluable for communication between people; this was, after all, the reason it evolved in our species. Despite the proliferation

of various nonspeech computer technologies aimed at communication, a series of studies in the mid-1970s by Chapanis and colleagues (Ochsman & Chapanis, 1974; Chapanis, 1975) suggest that voice is the most effective channel for problem solving. A large body of work in social psychology has explored the effectiveness of voice for problem solving and collaboration over the telephone; see Rutter (1987) for a review of the literature. Although more recent work (Minneman & Bly, 1991) offers insights into the utility of video accompanying voice for distant interactions, it does not challenge the dominance of voice. An extremely expressive medium, voice can be more subtle than written language. Chalfonte, Fish, and Kraut (1991) found evidence that voice annotation results in more effective teamwork for group writing; spoken comments were deeper and more substantive than written notes.

Speech is "natural" and ubiquitous; we carry our voice and hearing organs with us and learn to speak early in life. We employ elaborate strategies to insure that a listener understands our intent, and also to take turns while conversing. We talk and listen while engaged in other activities. We use speech to communicate at a distance independent of visual contact. Speaking is much faster than writing; we speak between 150 and 200 words per minute, while an experienced typist can produce less than half that amount of text, and handwriting is typically half again slower than the typist.

Although we can talk faster than we can input text manually, this time advantage does not transfer to information retrieval. Speed is one of the primary drawbacks of stored voice as a data type. We can easily read several times faster than listening to normal speech; thus there is an asymmetry between time spent authoring a message and listening to it. We experience irritation listening to a lengthy message on an answering machine or to a speaker who rambles instead of coming to the point. During a conversation, the listener takes an active role in moving the conversation along, using both visual and verbal cues, and sometimes interrupting, to keep up a satisfying pace. These "back channels" make interactive conversations much more productive (Kraut, Lewis, & Swezey, 1982).

A second major liability of speech is its *temporal* nature. Although the user's gaze can wander around a visual menu, a voice menu must be recited sequentially, item by item; the user must listen to every choice in order to get to the last one. Because the menu is transitory, a missed menu item can be accessed again only by repetition. The need for attentiveness increases the cognitive load on the user, who must remember the list of choices until a selection is made.

Speech is *bulky* compared to text. Speech recognition is not yet robust enough to support the search of files of stored spontaneous speech for keywords, an operation frequently performed on text. It is

harder to "skim" a voice file than a text file to find some desired piece of information, or even to determine whether a file is at all interesting. The act of listening requires our attention: while keyword searches of text files may take time, the user can do something else while they are in progress. Speech playback can be sped up by a factor of up to two or three using signal-processing techniques; simply playing samples back at a faster rate increases the pitch and makes the talker sound like the cartoon chipmunks.

Speech is *public*—it broadcasts through the air and can be heard by distant listeners. This can be either an advantage or a liability depending on the intended recipient of the voice message. Publicity is advantageous for alerting and asynchronous announcements, as it does not depend on the user's gaze being directed at a display. The public nature of speech is also essential for communicating to a large number of people: Airport flight announcements or emergency notification ("There is a fire . . .") should be heard by everyone. But some announcements are distracting, such as paging the recipient of a telephone call, as they interrupt everyone for the sake of getting a message to a particular person.

SPEECH TECHNOLOGIES

The preceding short discussion of the assets and liabilities of speech was made without regard to performance, i.e., how well the talker and listener actually communicate through words. Unfortunately, when one participant in the voice interchange is a computer, we encounter further limitations with the voice channel, as computers are imperfect listeners and talkers. This section will briefly review the speech technologies and their performance limitations.

Digitization

Sound is a continuously varying, or analog, signal. In order to be stored in computer memory, it must be digitized, which necessitates *sampling* it at some discrete time interval and *quantizing* it with a limited number of bits per sample value. In addition, the signal may be *compressed* to consume less storage or transmission bandwidth. How this process is accomplished affects the playback fidelity, or the difference between the original and reproduced signal.

Several factors influence the quality of sound digitization. The sampling rate imposes a bandwidth limitation on the signal that can be reproduced from the sampled original; the higher the sampling rate, the greater the frequency response.[1] A second factor is the resolution with which each sample is stored. Resolution limits how closely the regenerated waveform can match the original, and depends on the number of bits allocated to each digitized sample. "Telephone quality" speech is sampled 8,000 times per second at an effective resolution of 12 bits per sample.[2] The compact audiodisc is based on 16-bit samples at 44,100 samples per second.

The comparison between the telephone and CD illustrates how quality is proportional to the number of bits per unit of time employed to represent a sound. Sampling more often and with more bits per sample increases the fidelity of a digital recording, at a cost of increasing storage to capture the sound. Fortunately, telephone quality speech has acceptable intelligibility and at 64,00 bits per second is well within the capabilities of most current workstations. The data rate can be further compressed by taking advantage of knowledge about the spectral and temporal characteristics of the signal (i.e., speech versus music versus environmental noise). For speech, this can allow the data rate to be reduced by a factor of two with negligible loss of quality, or by a factor of ten with serious signal degradation but still preserving intelligibility. Such compression requires computation or signal processing at both record and playback time. For a thorough review of speech compression algorithms, the reader is referred to Rabiner and Schafer, (1978).

Many workstations and personal computers now include hardware to digitize and play back telephone quality speech. An increasing number also allow higher sampling rates, greater sample resolution, and stereo recording. Many speech compression algorithms can run in real time on today's microprocessors. Although audio digitization is not quite ubiquitous on computers, it is likely to become a standard capability without requiring additional hardware over the next several years.

Speech Synthesis

Text-to-speech synthesis allows the computer to speak text; an application can send words to the synthesizer, which produces a digital audio waveform to be played through a speaker or over the telephone. Speech synthesis is difficult and is usually performed in several steps. The first

[1] The Nyquist theorem defines the theoretical maximum frequency to be one half the sampling frequency. But real-world constraints of analog filtering on the input and output signals impose a somewhat lower frequency range.

[2] Digital telephone circuits actually employ 8 bit mu-law encoding, a logarithmic coder equivalent in dynamic range to 12 bits of linear coding.

step translates text into intermediate-level speech segments, typically phonemes, which have a consistent pronunciation. The second step of synthesis applies knowledge of the acoustic properties of phonemes to generate the proper sounds of speech. A final step assigns *intonation*, variations in pitch and duration of syllables, to give the resulting speech the proper rhythm and meter.

Unfortunately, errors arise at all stages of speech synthesis. English is a comparatively difficult language—the same group of letters can have many different pronunciations (e.g., *c* in *cat, cent,* and *chin*), leading to an incorrect choice of phonemes. Some words, such as *convict* and *read,* change pronunciation in syntactic context. Although many of the words with unusual pronunciation may be stored in a lexicon, the lexicon is unlikely to be large enough to store the pronunciation of all words. Proper names are particularly difficult to synthesize, because of their varying ethnic origins, each with its own text to sound rules. Names may also be especially difficult to understand if spoken incorrectly, because lexical context does not provide redundant cues for the identity of the name (Spiegel, 1985).

Even when no errors are made in letter to phoneme translation, the speech produced by the phoneme acoustic realization rules is of limited intelligibility. This is in part due to our incomplete knowledge of the full set of perceptually salient acoustical cues to phoneme identity, and in part to currently limited computational models of speech production. *Coarticulation* is the systematic modification of speech sounds as a function of the surrounding phonemes and indicates that synthesis must include analysis of words in groups, not just in isolation. The synthesized sentence may have an acceptable pronunciation but incorrect *prosody,* which also reduces intelligibility (McPeters & Tharp, 1984). Even more importantly, human speakers use acoustic stress to emphasize the most salient words in the sentence, but for the synthesizer to do so would necessitate full understanding of the sentence. Finally, when a human-authored passage is synthesized, typographic errors or incorrect punctuation interfere with listening comprehension much more severely than reading comprehension.

Synthesized speech may contain errors in choice of phonemes, and even correct phonemes are less intelligible than natural speech. This places greater cognitive demands on the listener to decode the speech, interfering with the user's ability to pay attention to the task at hand (Luce, Feustel, & Pisoni, 1983). On the positive side, though, listener comprehension of synthetic speech improves rapidly after short exposure, and this skill is retained even if exposure is infrequent (Pisoni et al., 1985). Although spelling and typographic errors detract from listener comprehension, sentence context aids the listener in understanding a synthesized passage.

Until recently, real time speech synthesis has usually been performed in specialized hardware attached to the computer either as an internal board or an external device. But processors have become fast enough to support all-software speech synthesis, taking advantage of audio hardware on the computer to play the digitized sound output of the synthesis process. This technological development will make synthetic speech much more widely available within the next several years.

Speech Recognition

Speech recognition is the least robust of any of the speech technologies considered in this chapter. The task of the recognizer is to listen to speech and identify the spoken words. Compensating for small variations in our speech is difficult, and recognizers are quite error-prone. All recognizers include a model of the words to be recognized, a means of capturing speech and converting it to the representation form of the model, and a pattern-matching function to determine which word is the closest match.

Recognizers may be differentiated along several dimensions. They may require a speaking style of *discrete* speech, with a pause between each word, accept short bursts of connected words, or be able to operate on the stream of *continuous* speech found in ordinary conversation. *Keyword-spotting* recognizers can identify a small number of words embedded in longer passages of continuous speech. Recognizers may be *speaker independent* and operate with any talker, or *dependent* on the particular user who trained the active vocabulary. *Speaker-adaptive* recognizers tune their vocabulary models to the user, without requiring explicit training of each word, but they must be given feedback for this learning to succeed. Finally, each recognizer is designed to operate on a limited vocabulary size from as few as two to over 25,000 words.

Recognition errors fall into three classes. *Rejection* errors occur if the recognizer cannot confidently match the user's speech with any word in its vocabulary. *Insertion* errors are the opposite; a word or nonspeech environmental sound which should not be recognized is mistakenly identified. *Substitution* errors occur when a word in the recognizer's vocabulary is spoken, but it is incorrectly identified as another word in the vocabulary.

A number of factors influence error rate. Several words in the vocabulary may be acoustically similar. Short words are harder to recognize than longer words because there is less acoustic information on which to base a judgment. Recognizers are notoriously susceptible to background noise or poor microphone placement. Some users have difficulty speaking consistently and clearly, which hampers recognition. As

the vocabulary size increases, recognition becomes more difficult because it is increasingly likely that several words sound similar. It is also more difficult to recognize connected speech than isolated words because of the effects of coarticulation.

As with synthesizers, recognizers to date are usually implemented as additional hardware to attach to a computer. Progress towards all-software speech recognition has been slower due to the need to perform computationally intensive acoustic processing on the incoming speech signal. Although very limited recognition is possible in software utilizing workstation digitizers, more useful recognition is likely to require an associated digital signal processor for some years.

APPLICATION ENVIRONMENTS

Although speech is our most powerful medium of communication, it is awkward and demanding as a computer data type. Limitations in speech technologies hamper their utility as user interface components. As a result, speech systems have not yet been widely deployed, but instead are applied to limited or very specialized application areas.

For example, speech synthesis has been employed as an aid to the disabled; indeed, much of the early work in text-to-speech synthesis was aimed at developing a reading machine for the blind. Speech recognition is also used successfully by the disabled, allowing the motor-impaired to control computers and mechanical equipment. Recognition has also found a niche in industrial and laboratory environments where a user's hands and eyes are busy performing some other task, such as inspecting printed circuit boards or appliances on an assembly line, sorting airline baggage, or entering results of visual examination of laboratory specimens while using a microscope. In all of these applications, speech technology is employed because it offers a distinct advantage to an otherwise disadvantaged user for whom conventional computer interfaces are inadequate.

The apparent success of voice mail indicates that digitized speech is immediately useful for a much larger user population. Digitized voice accessed over the telephone is also the basis of a growing variety of interactive information retrieval applications, such as bank balance inquiries, flight information from airlines, and nationwide weather forecast services. The success and sheer number of such services indicate the willingness of many users to tolerate the slower speed of speech in return for the ubiquitous availability of the telephone. But these telephone-based systems are limited in that the information is presented in a single mode by the service provider, and accessible only over the telephone. From the user's perspective, such applications are

part of the telephone network, and thereby divorced from the variety of personal information systems and databases in use on personal computers.

The rest of this chapter considers the evolution of speech technologies into the very different world of our every day work environments. With the emergence of powerful workstations and increasing computerization of business and communication, speech has renewed potential to reshape our interactions with office technology. The claim presented here is that voice can offer enticing new ways of interacting with computers across varying distances; mobility is the essence of many speech applications. To support this claim, we must consider, not only which classes of activity can benefit from the various technologies, but also what interaction techniques will facilitate their acceptance.

To this end, we will consider three different styles of interaction across work environments employing dramatically varied physical affordances to voice technology. Because of limitations in speech and speech technologies, successful voice applications will emphasize the unique aspects of voice, while accounting for and perhaps exploiting the user's physical environment. Although this chapter makes no pretense of *predicting* the voice market, it will suggest *potential* avenues for breakthroughs towards widespread acceptance of voice technologies. It offers examples of such applications in the hope of stimulating further exploration of the creative use of the voice medium.

AT THE WORKSTATION

The first work environment to consider is the most common one today: a person sitting at a desk in front of a computer display, keyboard, and mouse. While much attention has been paid to speech recognition to command the computer and enter text, this is but one use of a single speech technology. Voice, both input and output, has a much broader potential at the desktop. It is important to consider voice in an integrated context both for the user interface as well as a data type. This unified view of voice as a workstation resource may be labeled *desktop audio*, and spans applications, user interfaces, and system architectures (Schmandt & Arons, 1989).

Speech Input

From user interface designers to science fiction writers, the most popular "vision" of desktop speech processing features the concept of talking to one's computer. Such an interface can take many forms, from dictation of a text document, to application-specific voice dialogues, to

voice access of the operating system or window manager. Each of these has very different requirements of speech recognition technology.

The *listening typewriter,* which automatically transcribes voice into text, is being pursued by many research and development groups. Encouraged by a series of studies by Gould (1982; Gould & Boies, 1978), which indicated that dictation could be learned quickly and used effectively, the developers of the listening typewriter foresee its widespread acceptance. But caution is in order; a high-quality listening typewriter is still a distant goal of researchers, despite the claims of some current commercial ventures. Its utility is limited in several ways; in the near future, less powerful and more readily available recognizers can have an immediate impact on a larger user population.

Although seemingly generic, a listening typewriter implementation is a specialized application of speech recognition. Designed to understand common English prose, the listening typewriter may not perform well in other tasks, and thus may not be applicable for tasks other than writing, such as controlling particular applications or invoking operating system commands by voice. In order to recognize large vocabularies with satisfactory accuracy (20,000 or 30,000 words is a popular goal at the moment) it is necessary to employ constraints in the language. For example, one recognizer uses a language model based on the probabilities of sequences of multiple words (Jelinek, 1985) to augment the acoustic evidence of which word was spoken. Another recognizer requires the user to look at a display to confirm that the correct word was recognized before proceeding to the next word; if an erroneous recognition is not first corrected, the recognition software makes false assumptions about the probabilities of following words.

Such language models, based on business correspondence, may not be suitable for other uses. Although many tasks are very structured, at certain points, such as naming a file or directory, the branching factor, or *perplexity,* of the language model becomes very high, making it extremely difficult to recognize the next word. Because of their terseness, system commands contain little information which is redundant across words; recognition of each word is critical. File names can be spelled letter by letter, but this is slow, and as will be discussed momentarily, spelling recognition is particularly difficult.

Dictation with speech recognition may not be much faster than typing. Most of the current large vocabulary speech recognition systems use isolated speech, which is slow and cumbersome.[3] In addition, one-

[3] Although Gould (1983) claimed that users were neither slower nor less satisfied with isolated word recognition as compared to connected recognition, this could be an artifact of his experiment. Because users could not correct words except by rubbing out everything from the current entry point back to the word in question, they probably entered text word by word or in small groups even in the connected speech mode.

half to two-thirds of the time spent composing a letter is incurred in planning (Allen, 1983); this limits the net speed contribution of faster text entry. Many documents require additional editing, which may consume more time than the original typing task.

Although a large vocabulary listening typewriter will dramatically change word processing for particular user populations (those unable or unwilling to type), it poses a distant promise for revolutionizing the way most of us interact with computers.[4] However, smaller vocabulary recognition, which is currently practical, can be adapted to specific applications with high degrees of success (Lee & Hon, 1988). Not surprisingly, the most promising applications are those already overloading manual input, such as computer aided design and drawing packages.

In applications in which a mouse or other graphical input device does double duty, both to indicate position as well as to manipulate menus, speech recognition can augment the mouse with voice access to menus. In addition, divided attention theories (Allport, Antonis, & Reynolds, 1972; Wickens, Mountford, & Schreiner, 1981) suggest that splitting the two tasks across two input modes will increase user performance. The validity of these points was demonstrated in an experiment using voice recognition in a circuit design task (Martin, 1989); although based on somewhat limited experimental data, this work offers a good overview of user behavior using speech input.

Application-specific speech recognition is viable with current technology, provided that either the application can be controlled by a limited input vocabulary or the structure of its interaction constrains the number of possible word choices at any moment. Voice input is useful for filling out forms, for example, as each juncture in the interaction is focused on a particular vocabulary subset. An example of a successful application in this arena is radiology reporting, in which a physician dictates a report while viewing X-rays. Although seemingly free-form dictation, this application actually takes advantage of the limited perplexity at any point in filling out a report using a standard format.

Application-specific speech recognition has limitations in the context of general purpose workstations. With the advent of window-systems and the dominance of multiprocess operating systems, users run many applications simultaneously on a single screen; will they all compete for the microphone? Although an audio server (Schmandt & Arons, 1989; Angebranndt, Hyde, Luong, Siravara, & Schmandt, 1991; Schmandt & McKenna, 1988) can allow multiple client processes to

[4] This claim is not meant to imply that the *research* on large vocabulary speech recognition is not of major importance to the remainder of recognition development.

share limited physical resources, a mechanism still must be provided to allow the user to shift attention, and thereby choose which application should listen to the microphone input.

Focusing on the widespread use of windows, a Media Lab project, Xspeak (Schmandt, Ackerman, & Hindus, 1990; Schmandt, Hindus, Ackerman, & Manandhar, 1990) explored use of small vocabulary recognition to switch between windows and invoke applications under the X window system. Xspeak allowed users to "name" windows; upon speaking a window's name, the window would move to the foreground becoming completely exposed, and the cursor would immediately appear in the named window to receive keyboard input.[5] Xspeak was motivated by the desire for a hands-free means of manipulating a large number of windows, and evaluated over several months of use by student programmers. For users of a small number of windows, or those who had already developed techniques for dealing with multiple windows, such as icon managers or "rooms" (Card & Henderson, 1987) of windows, voice input provided marginal added value. Voice was attractive to all other users, and they employed it using a variety of techniques, but complained that it seemed silly to invoke an application by voice but then have to type or mouse at it.

A motivating factor behind Xspeak is the mismatch between the two and a half dimensional world of overlapping windows and the two dimensional nature of the mouse. As windows proliferate, finding them with the mouse becomes increasingly difficult, while recall by voice is constant (subject to the ability of the user to recall one member from a list of names). We noted that the mouse was actually faster than speech for the most simple tasks (although users had the opposite impression), but voice shows strong performance improvements for more complex window layouts and operations.

Xspeak suffered from poor recognition quality, even though it used a small vocabulary with isolated word input to simplify recognition. Much of the degraded recognition was due to our unwillingness to use head-mounted noise-canceling microphones; instead we used a microphone positioned next to the workstation monitor. Although noise-canceling microphones result in much improved recognition, the user must not be forced to wear an uncomfortable piece of equipment that is not acceptable in most office environments. This limitation is prevalent in currently available recognizers, and improved speech recognition

[5] Focus management was actually the responsibility of the window manager. Xspeak was not a window manager, but just another client of the X server. It operated by mapping window IDs to the recognizer's vocabulary, modifying the window stacking order for visibility, and warping the mouse pointer.

under imperfect acoustic environments is one of the main challenges of current recognition research.

Applications were not modified to use recognition under Xspeak. An alternative approach to recognition at the desktop is to allow voice to be used across multiple applications, with each application managing the recognition results much as each window receives and interprets its own mouse and key press events. The OM (Office Manager) system from CMU (Rudnicky, Lunati, & Franz, 1991) employs a "voice manager" to direct speech input to several applications, including a personal schedule, name and address database, and calculator. Both *focus* (i.e., determining which application receives voice input) and *attention* (i.e., acting upon or ignoring any recognized words) are controlled by mouse and voice. OM uses a "click to talk" strategy to allow the user to activate speech recognition. Explicitly specifying the application to receive voice input can improve recognition performance as it limits the set of possible utterances at that point in time.

An older Media Lab project, Conversational Desktop (Schmandt, Arons, & Simmons, 1985), also used voice input to control a number of applications, including telephone dialing, schedule management, and database queries. This system used an implicit focus mechanism; each request from the user was routed to the appropriate application action routines based on parsing the contents of the request itself. Attention was managed by taking advantage of the directionality of human speech; recognition was enabled only when the user spoke while turned toward the microphone.[6]

These examples also illustrate that adding voice to an application is not as simple as writing macros to simulate mouse motion and keystrokes to manipulate an underlying application menu structure. The likelihood of errors in speech recognition necessitates dialogue about what the user said in order to do what the user wants. Since we talk for many reasons in an office, an attention mechanism is required to prevent a steady stream of insertion errors. If the recognizer's acceptance threshold is set very high to try to avoid the need for explicit indication of input, rejection errors are multiplied. Errors are more likely and more complex with connected recognition, which was employed in both the example systems just mentioned. Both employ dialogue strategies incorporating very different interaction techniques.

OM displays recognition results as text in a window. The user can then click on misrecognized words, and type them or speak them again. This affords a simple method for the user to detect and correct recogni-

[6] The amplitude of the speech arriving at each of an array of microphones was sampled and used to control audio input to the recognizer.

tion errors, and simplifies the language understanding burden on the application. But this technique distracts from the advantages which might be gained by using voice, i.e., freeing the user's visual attention and manual activity for other tasks. Speech input could allow users to control one application while using another with the keyboard and mouse, but an alternative dialogue strategy is necessary.

Conversational Desktop used voice response in a rather different discourse strategy. Recognition results were analyzed by a parser (described in Schmandt & Arons, 1986) capable of parsing incomplete utterances or those in which substitution errors occurred. At each step in the parse, a record was made of what additional information would be required to complete the sentence. For example, if the first word was *call*, the parser would note the requirement of the name of someone to call. If a name was encountered later in the sentence, this requirement would be removed.

Once all input had been exhausted, the list of missing information revealed whether any recognition errors had occurred.[7] The list of missing items was combined with the recognized words to generate a carefully selected spoken query, such as "whom do you wish to call?" Conversational Desktop listened for additional input, which was merged with what had already been parsed, and the process continued.

This spoken exchange allowed the user to perform the speech task while keeping hands and eyes free. But spoken dialogue has its own difficulties (Hayes & Reddy, 1983), which are confounded if the user's responses are misrecognized as well. Even though speech output is slow, it is important to echo much of the user's input to ensure against substitution errors, even though this further taxes the user's attention. The demands of the various input modalities and dialogue techniques must be balanced against the attentional characteristics of the applications in which they are employed.

In summary, speech recognition at the workstation is problematic. A large-vocabulary listening typewriter could be a boon for word processing, but it remains an elusive goal and, more importantly, is not a panacea that will make the keyboard obsolete. Application-specific speech recognition is more practical, and may afford significant improvement for selected applications, but potentially complex discourse techniques must be employed to cope with errors; adding voice input to an application usually requires rewriting the application. In an environment supporting a multitude of applications, speech may be used to choose between applications, or to interact with several applications

[7] Note, however, that semantically correct substitution errors, such as "call Jim" for "call Kim," could not be detected using this method.

in sequence, although there are some challenging architectural and user interface issues in managing this interaction. Finally, obtaining high quality speech recognition without encumbering the user with head mounted microphones is an open challenge which must be met before recognition will be accepted for widespread use.

Speech Output

There are two main roles for speech output, either digitized or synthesized, when used at the workstation. The first is recorded *voice messages*, such as a voice mail, annotations to a text document, or part of a multimedia presentation. These provide voice as a data type and require digitized speech. The second role is *alerting* the user to an event, such as the arrival of new electronic mail, a shutdown message from system administration, or an alarm based on the user's schedule. Such notification can employ synthesized speech (for example, to announce the sender and subject line of the electronic mail), or digitized sound of either speech or nonspeech audio.

Stored voice can be used as a data type in a number of ways. The voice mail explosion demonstrates the utility of telephone messaging; voice is an effective medium for messages which are short, casual, or short lived. Providing access to telephone messages on a workstation screen allows enhancements such as improved interfaces and the ability to annotate a message with text either for archiving or to forward to another user. A voice mail retrieval tool may communicate with other workstation applications, such as a telephone dialer or personal address book, to make it easier to reply to a message (Stifelman, 1991; Kamel, Emami, & Eckert, 1990).

Because speaking is faster than typing, workstation users may prefer sending some messages as voice rather than text (although most recipients would prefer reading to listening). But when workstation-based voice messaging has been made available in work environments accustomed to electronic mail, it has certainly not displaced text messages (Nicholson, 1985). This suggests that authoring voice messages is more important to telephone-based interaction, while the workstation may offer the most improvement in voice message retrieval.

Voice is powerful when used as a data type in conjunction with other media. Voice is powerful for annotating other media, in part because it can afford an added dimension to the flat world of text and graphics. As noted in the previously mentioned work by Chalfonte et al. (1991), voice can be a richer and more effective medium for communicating comments on a document under review. Finally, voice is an essential component of multimedia presentation systems. For example, Zell-

weger's (1989) Scripted Documents allow voice segments to be played synchronously with text display.

Voice can be used as a document type on its own, although browsing a voice document presents several difficulties. Muller (Muller & Daniel, 1990) suggests a hypertext-like voice document architecture, employing short linear sequences of speech and allowing the user to choose a new path through the document at each juncture. But because voice is a time-varying medium, it is sometimes difficult to determine at exactly which juncture the user made a selection, due to variable user response times. In addition, voice as a document type, or as a recording of events such as lectures and meetings, begs for a user interface supporting audio scanning or time compression techniques to allow the sound to be played back much more rapidly than it was recorded.

In a Media Lab project, Hyperspeech, Arons (1991) explored the potential for a voice-only hypermedia system by implementing a prototype. The document consists of recorded reviews with five experts on user interface design. These interviews were segmented into short sequences, with links describing routes from each sequence to the next, just as with conventional hypertext systems. A user navigates the database using speech recognition, and the system responds to queries such as "Who said that" using synthesized speech. The effect is described as *having a conversation*, and establishes one's sense of the interviewees personalities to a greater extent than could have been afforded by text alone.

When accessing stored voice at the workstation, a graphical user interface helps overcome some of the limitations of speech output, namely its slow and serial nature. A visual user interface can indicate the presence of a voice segment, display its duration, and provide a direct manipulation means of starting and stopping at various places within the segment. Figure 1 illustrates several graphical interfaces, ranging from a simple push-to-play to those which provide more flexible access by allowing the user to play part of a sound.

The utility of voice as a data type can be considerably increased by adding the ability to move it between applications. Window systems allow text to be selected from one window and inserted into another, and similar interaction techniques can be employed with graphical representations of voice. Figure 2 shows several Media Lab applications supporting *audio cut-and-paste*. On the left is the visual user interface to voice mail, and on the right is a calendar supporting both voice and text as data types; the user can select part of a voice message and insert it in the calendar as an appointment.

If we make a mistake while recording only a few seconds of speech,

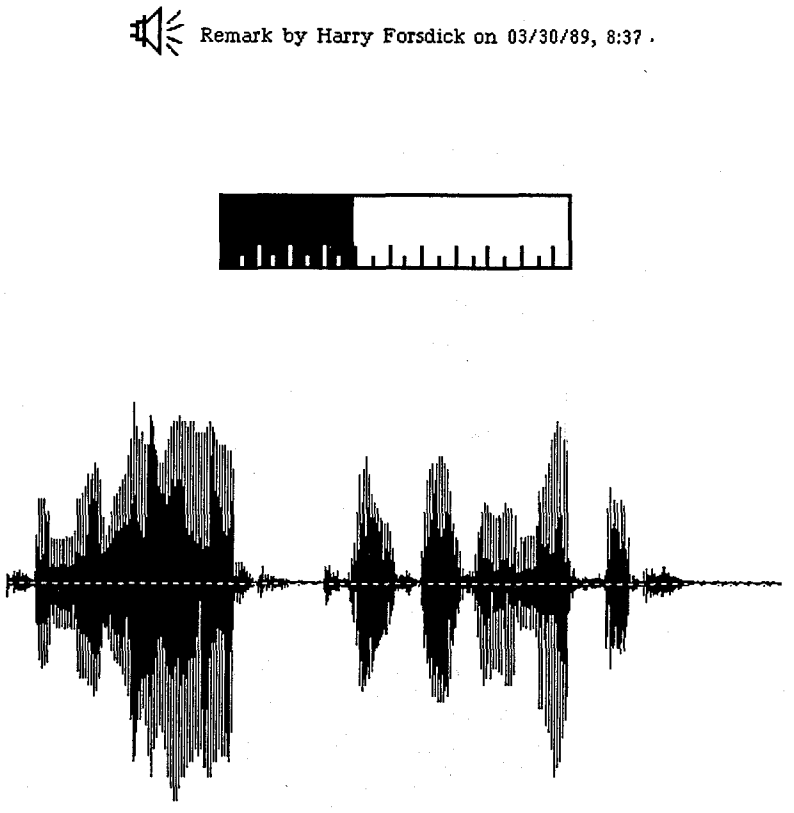


Figure 1. Several methods of representing stored sounds. At the top is an icon from BBN's Slate multimedia document product. In the middle is the Media Lab's SoundViewer widget. The representation at the bottom is a sound waveform envelope from Mixview, a public domain audio editor.

it is easy to simply start over from scratch. When authoring longer text messages, users will require voice editing. An audio editor provides a visual representation of the digitized sound, and a graphical user interface for editing. Although some editors display the sound waveform, a simple representation indicating intervals of speech and silence may suffice as a navigational aid (Ades & Swinehart, 1986). Although users of a voice editor have been reported by Allen (1983) to desire to edit individual words, this is impractical due to the continuous nature of speech. Inserting or deleting units of speech smaller than a sentence or

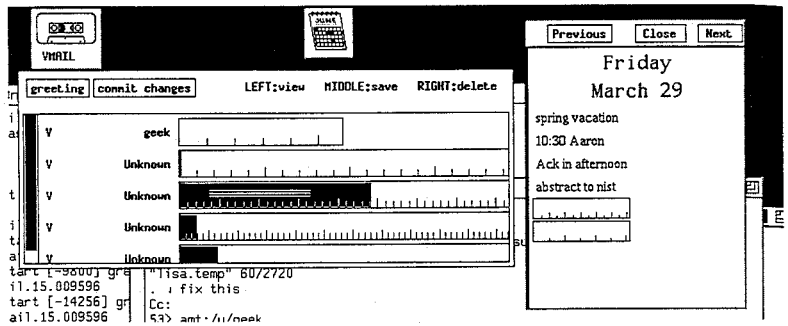
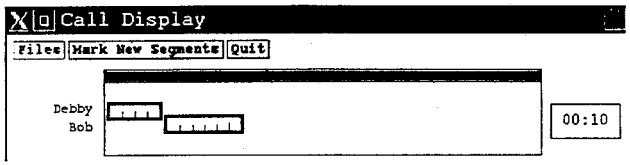


Figure 2. Audio cut-and-paste between voice mail and calendar applications. On the left is the display of incoming voice messages. On the right is a calendar application which supports voice as well as text. Using the mouse, voice messages can be copied into the calendar.

phrase results in poor prosody and sounds choppy. Finer grained editing is only useful during preparation of a recording destined for eventual transcription; consequently, the audio discontinuities are irrelevant.

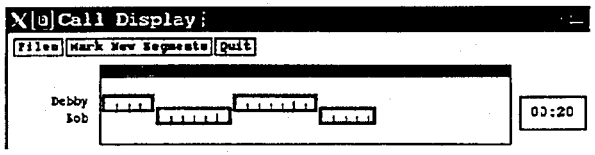
The applications discussed for stored voice make use of small snippets of intentionally made recordings. But one of the most powerful aspects of speech is the degree to which we depend on it in our daily work lives. If computers could capture some of this speech (relatively easy) and more importantly present it for later retrieval in a helpful manner (more difficult) they could become powerful assistants. Work at EuroPARC (Lamming & Newman, 1991) tries to correlate audio and video recordings with the location and other activities of a community of users. Recent work at the Media Laboratory (Hindus, 1992) focuses on the more restricted domain of telephone calls, by incorporating a capture tool for use during the conversation, and a browser for later retrieval (see Figure 3). During capture, conversation structure is derived from the available acoustic evidence (pauses and turn taking) and presented at the user interface to facilitate selection of a portion of the conversation to be saved.

So far this section has focused on applications and interfaces for stored voice as a medium for a message from one person to another, or a message to oneself for later access (e.g., voice in the calendar). In addition to voice as a data type, audio can be used for alerting messages from the workstation. Combinations of synthesized speech (for content) and distinctive digitized sounds (to distinguish message type) offer opportunities for increased aural communication to the user over the ubiquitous computer terminal "beep." Audio notification can alert the



Debby: Hello, this is Debby Hindus speaking.

Bob: 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.



Debby: Well, I think it'll take me about another hour, hour and a half, to finish up the things I'm doing now.

Bob: OK, I'm just going to head on home, I'll probably do a little shopping on the way.

Figure 3. A visual interface presents the structure of a telephone conversation to help the user decide which portions to save.

user even when not paying attention to the workstation screen.[8] Applications using windows which are obscured or off-screen may choose audio over visual messages.

Synthetic speech can be used to announce the arrival of incoming electronic mail, including author and subject, as well as voice mail, faxes, and other forms of electronic communication. At the Media Lab, speech synthesis is used for paging with phone calls, and also as a substitute for the telephone bell in offices; for on-campus calls the caller's name is included in the announcement. Voice announcements can be disrupting, however, so such applications are likely to be more beneficial if they employ *filtering* based on user-defined rules specifying which messages to announce.[9] As the use of synthesized speech

[8] On the other hand, if the user is not present when a voice announcement occurs, the information is lost. A combination of aural and visual notification may be optimal.

[9] The telephone bell is an example of unfiltered audio announcements.

increases, users will be less offended by its poor quality simply because they can so readily learn to understand it (Pisoni et al., 1985), and they may configure workstation announcements much as they personalize the screen layout of their windows.

Nonspeech audio can also provide useful cues in the user interface at workstations. Gaver's "Sonic Finder" (1986) added sound as a status indication in various system operations invoked by the user; although anecdotal, he reported some evidence that users employed the sounds to help keep track of what they were doing. In more recent work, Gaver found sound output helped multiple users collaborate on a process control task spanning two workstations; although neither user could see both screens, both users could hear the sounds emanating from the other workstation (Gaver, Smith, & O'Shea, 1991). Nonspeech sounds may place less cognitive load on the user because they can be much more brief and distinctive than voice messages.

Speech output, then, can be employed at the workstation in a number of ways. It affords new utility to stored voice as a data type; a graphical user interface removes some of the obstacles to employing this rich means of conveying messages. Digitized speech can be supported either as an adjunct to text and other media, or with somewhat more difficulty, as the primary document medium. Both synthesized speech and digitized sounds can be used for alerting and announcements of many sorts.

The viability of voice output at the workstation is much less ambiguous than for speech input. This is due in no small part to the fact that the output technologies are much more robust than speech recognition, at least for the next 5 years. But equally important is the already widespread acceptance of stored voice messages as voice mail or on answering machines; we are becoming increasingly comfortable speaking when only a machine is listening. This section has proposed that workstation-based interfaces offer enhanced functionality and ease of use over telephone-based access to voice mail. The next section will suggest that the telephone may become the primary source of voice as a computer data type.

It is unlikely that any single application just described will revolutionize how we compute in the office. But desktop audio architectures can allow multiple applications to share audio resources such as speakers and microphones through an audio server much as applications share the screen, keyboard, and mouse under a window system. The operating system can provide dynamic allocation of process resources, so that speech recognition, synthesis, and compression can be provided with little additional hardware cost. Applications written to provide compatible data exchange facilitate the acceptance of voice as a data

type in many desktop activities. The workstation environment is rich in opportunities to exploit voice in everyday work activities.

The discussion of voice at the workstation would be incomplete without mention of "noise pollution" in the workplace. Sound travels through the air and easily disturbs others. Earphones and noise canceling microphones worn on the head can alleviate these problems, but many potential users are justifiably averse to such encumbrance. The telephone handset provides a partial solution (e.g., occasional private listening to voice mail messages, or infrequent recording of voice memos) but is at cross purposes to notions of speech recognition offering an alternate input channel in hands and eyes busy user scenarios, or of voice alerting gaining the user's attention during other activities. The physical layout of office space may be as critical as voice technology itself in the workstation environment.

TELEPHONE ACCESS TO WORKSTATIONS

A second environment in which speech has potential for computer interfaces is remote telephone access to applications. As an alternative to logging in with a terminal and modem, a user of an interactive telephone interface drives applications with speech or touch tone input, while the application responds with synthesized or recorded speech. Telephone access to voice mail is currently the most common example of such interfaces, but a much wider range of personalized services can be supported. Because telephone-based applications cannot take advantage of a display, they must use voice both as the data as well as the interaction medium. This renders these applications doubly susceptible to all the difficulties associated with speech, and the situation is further confounded by some limitations imposed by the telephone connection.

The main benefit provided by telephone-based voice applications is remote access; the application can be used from any telephone without additional equipment. In order for an application to succeed, the advantage gained from improved access must offset the difficulty of using a voice-only user interface and presentation medium. In this environment, voice input and output go hand in hand, so this section will focus on voice-only interaction techniques and offer examples of telephone-based services which make workstation databases available at a distance.

Speech recognition over the telephone network is especially problematic. Telephone lines have a limited bandwidth, which removes some higher frequency information and thus some of the cues that help

us understand speech. Telephones offer a limited dynamic range for the amplitude of voice signals. Telephone connections may be noisy, especially when calling on telephones from public environments such as airports. Cellular telephones provide increased mobility, but current cellular systems provide notoriously poor audio quality. All of these factors make recognition difficult.

Another problem with voice input is the nature of the audio signal path in a telephone connection. The telephone provides a single electrical circuit, which carries the speech of both parties. Each telephone set is equipped to remove much of the transmitted signal from the received signal so as to minimize the amount that we hear ourselves talk, but this subtraction is imperfect. Because it is difficult for a speech recognizer to sort out the two sides of the conversation, the user is prevented from speaking while the computer application is talking. But interruption is an essential technique for speeding up a conversational interaction, as it helps compensate for the slow nature of speech.

One solution to this problem is an echo canceler, which determines the acoustic characteristics of the telephone line and performs the signal processing required to accurately separate the two sides of the conversation. A less elaborate alternative is to monitor the incoming side of the telephone call only during the pauses between phrases and sentences in the computer's speech. If any sound is heard during these times, the application can stop speaking and begin recognition. The caller must force the interruption, however, perhaps saying, "Uh . . . next message," to get a word in.

Speech recognition over the telephone network faces all the performance problems discussed earlier confounded by the acoustic limitations of the telephone; today it is in limited use. The common alternative to speech recognition input is touch tone signals.[10] Touch tones are loud and contain distinctive frequencies, so they are usually detectable through speech, and hence can be used to interrupt. If detected, tones are readily and correctly identified. Callers are experienced with telephone keypads for dialing and entering credit card information, and now are increasingly likely to take advantage of telephone-based services such as bank account transactions, train schedules, or weather forecasts around the country. Touch tone interfaces can be adopted as a remote interface to the computer at one's office.

Speech output and touch tone input are combined to provide audio menus, e.g., "For weather information press one, for traffic reports

[10] Technically referred to as DTMF (dual-tone multifrequency signalling), because each tone is a mix of two base frequencies. One frequency corresponds to the row and the other to the column of the telephone keypad.

press two. . . ." Voice menus require careful design (Englebeck & Roberts, 1989; Resnick & Virzi, 1992) to accommodate the slow and serial nature of speech. The user must concentrate on the menu during its presentation, because if the desired option is not understood, the entire menu must be repeated. Key design issues are menu size, prompts, and selection ordering. If the menu is interruptible, the experienced user can "type ahead" and make selections before hearing more than a few words of the menu, or any of the menu at all.

Another form of user input is selection from a list of known *strings*, e.g., names in an address book or file names in a directory. In North America, letters (except Q and Z) appear on the telephone keypad, allowing the caller to respond to a prompt such as "Spell out the last name of the party with whom you wish to speak," instead of speaking with a receptionist. Because each key is associated with three letters, a keypress sequence does not specify a unique alphabetic string: for example, the sequence "2 2 3" could spell either *bad* or *ace*. Although the interface could require two keypresses per letter to identify each letter uniquely, this is usually not necessary and does not even solve all spelling problems. As pointed out by Davis (1991), for many common lists (e.g., people's surnames, street names) the confusability between members of the list attributed to touch tone letter mapping is rather small; the collision on *bad* and *ace* would be a problem only if both were choices in a task at hand.

Even if the touch tone to letter mapping causes no confusion, multiple members of a list may have identical spellings, e.g., the set of people named *Smith* or the set of streets named *Cambridge* in the municipalities in the Boston area. This necessitates a method of selecting one among a number of matches, such as "If you mean Cambridge Street in Somerville, press one, or for Boston press two." Because the application must support selection by menu even when in spelling input mode, it is simple to allow the user to type in only a few letters, and then present a menu of possible completions when the user stops typing or presses a special key, such as "*".

Speech recognition faces similar problems when it is used for spelling. A number of English letters sound very similar, making it extremely difficult to distinguish between them. The *E set* is the largest, with *b, c, d, e, g, p, t,* and *v,* but there are several additional confusable classes, such as *a, j,* and *k.* If the recognizer cannot reliably distinguish between members of each set, then the spelling algorithm must be similar to that for touch tones; it must rely on both the size of the data list to limit possible confusions as well as explicit selection by the user when collisions occur.

Once a telephone-based application has determined what the user

wishes, it responds by providing some desired information. If this information consists of recorded speech, e.g., a voice mail message, it must be presented by playing the speech. If the information is human-authored text, such as an email message, speech synthesis is required. In many applications the data is extracted from a database and either recorded or synthesized speech can be used to speak it. If the range of possible values for the data is limited, small segments of recorded speech can be pieced together; however, the segments of digitized speech to be concatenated must be recorded with great care to ensure a smooth sequence of phrases. Concatenation invariably suffers from obviously incorrect prosody. Although synthesized speech is more flexible than recorded speech it is more difficult to understand by the inexperienced user.

Many existing telephone-based applications provide public information and are pitched at infrequent users. Similar telephone-based user interfaces can be used for applications allowing access to personal databases which would otherwise be available only at the office. A frequent user of telephone-based applications can utilize interfaces tuned for terse interactions. The Speech Filing System from IBM (Gould & Boies, 1984) was an early example of using the telephone to manage voice recordings for a variety of applications. The addition of speech synthesis allows voice and text to be mixed freely as items in the database. Some candidate applications for multimedia access include voice mail, electronic (text) mail, calendar management, and name lookup from an address book. An example from recent work at the Media Laboratory is illustrative of telephone access of personalized information.

The "Phoneshell" lets users log in from a touch tone phone and access voice mail, a calendar, a personal address book, dial-by-name from the lab staff list, and electronic mail. During one session, the user can move between applications via a top-level menu. Phoneshell is not meant to replace graphical user interfaces, but rather to offer access to some important databases when a graphics display is not available. Phoneshell uses touch tones for input and both recorded and synthesized speech for output.

In some ways the voice mail system is similar to commercial products. Much of its novelty lies in the graphical user interface shown in Figure 2, which is provided by a different program. But several differences illustrate the power of integration, both across interface media as well as across applications (Stifelman, 1991). In addition to being able to record messages to other voice mail subscribers, a user can record a personal memo. The memo is a message to oneself presented only with the graphical user interface; it is a reminder to do something

"back at the office" and becomes part of a personal things-to-do list. Voice mail messages can not only be forwarded to other subscribers but also sent to other applications. To move a message into one's calendar, the user specifies a month and day with touch tones.

The calendar application (Schmandt, 1990) uses speech synthesis to recite entries in one's calendar; touch tones are used to specify a day and month and navigate among calendar entries. The calendar is an effective example of the utility of remote access, as a calendar user must always consult the calendar before scheduling an activity, but the new activity must then be added to the database. To add to the calendar, the caller picks a day with touch tones and then records the entry. As a result, the calendar database supports voice and text, which requires the graphical user interface to support voice as well. A portion of this interface can be seen in Figure 2; it employs the same representation for stored voice as the voice mail viewer.

The address book application, *rolotalk*, allows the user to spell out a name and retrieve information including phone number, home and work address, and electronic mail address. Alternate search strategies can be invoked to search on first name, company name, or last name. Once a person has been selected, rolotalk places a conference call to that person, and listens in for a few seconds to allow the caller to cancel the call and return to the application. This application illustrates how context can be used to improve the intelligibility of speech synthesis by use of specialized text to phoneme rules. For example, the last four digits of my phone number are *not* "five thousand one hundred and fifty six," which is how the synthesizer would pronounce "5156." Similarly, the periods in my internet address (media.mit.edu) are pronounced "dot," and my street address is in "Massachusetts," not "MA." Context sensitive text preprocessing allows text fields such as these to be spoken correctly.

A third application under Phoneshell is *mailtalk*, an interface to reading electronic mail by speech synthesis; a previous version of this application was described by Schmandt (1984). Reading mail over the telephone is quite taxing, due to both the slow pace of synthetic speech and the cognitive load it places on the listener. Creative strategies are needed to respond to text mail in the absence of a keyboard. Nonetheless, it is possible to provide adequate functionality to allow a mail subscriber to dispose of or respond to many mail messages from any telephone.

Two techniques aid in reducing the amount of material presented to the caller and the difficulty understanding it: filtering and presentation strategies. Filtering at the message level applies rules based on sender, subject, recipients, etc., to decide whether a message should be spoken

```
Received: by media-lab.media.mit.edu (5.57/DA1.0.3)
id AA 19166; Thu, 27 Jun 91 15:30:43 EDT
Received: by inet-gw-1.pa.dec.com; id AA06824; Thu, 27 Jun 19 12:29:22-0700
Received: by gilroy.pa.dec.com (5.57/4.7.34)
id AA20087; Thu, 27 Jun 91 12:29:20 PDT
Received: by piglet.pa.dec.com (5.57/4.7.34)
id AA24720; Thu, 27 Jun 91 12:29:19-07000
Message-Id: <9106271929.AA2470@piglet.pa.dec.com>
To: Chris Schmandt <geek@media-lab.media.mit.edu>
Subject: Re: meeting in July
In-Reply-To: Your message of Thu, 27 Jun 91 14:34:06-0400.
             <9106271834.AA16114@media-lab.media.mit.edu>
Date: Thu, 27 Jun 91 12:29:18 PDT
From: Susan Angebranndt <susan@Pa.dec.com>

Morning of the 8th is okey with me.
```

Figure 4. The header may contain more text than the body of an electronic mail message.

or saved for eventual screen access. Information Lens (Malone, Grant, Lai, Rao, & Rosenblitt, 1987) was an example of rule-based mail filtering; such approaches are becoming more common for text-based mail readers. Filtering also must be used when reading a message, since a significant portion of a message is irrelevant header information (Figure 4). An even greater challenge for voice access is filtering message contents. For example, it is common for a reply to include some or all of the original message, often indented or preceded by a special character such as ">" on each line. An electronic mail reader could preface such a section by announcing "Included text follows." and allowing the user to skip over the entire included section with a single keypress.

Some of the information to be presented can be made more comprehensible by either preprocessing the text to be spoken, or understanding and translating it to different terms. Our mail reader tries to identify the real name of the sender, and precedes each message with this name and the subject line. A "more information" command expands these details, including the sender's network address and time of message receipt. The address of the sender is preprocessed, just as in rolotalk, to cope with the idiosyncratic pronunciation of these fields. References to time are translated from the literal to elapsed time;[11] Mailtalk speaks with greater specificity about more recent time. For example, in response to inquiries made at various times, "14:27:13" translates to "about half an hour ago" at 3 o'clock, or "yesterday mid-afternoon" on the next day, or "last Tuesday" a week later. The point of this conversion is to speak only the information that is most salient in

[11] This is especially useful when traveling in different time zones.

the context of the user's request, both to minimize speaking time and, more importantly, to minimize the cognitive load on the user.

One aspect of mailtalk, sending a reply, illustrates the synergy between the interaction environments being discussed in this chapter. The user has an option of recording a voice reply to the sender of an electronic mail message; this reply is sent as an audio file attached to, and sent by, ordinary electronic mail. Sending a reply is vital if the mailtalk user is to be able to act on the information in a message; the alternative is listening to messages and then making phone calls, if the phone numbers of the senders are known. Mailtalk supports several formats of multimedia messages, including those recently promulgated by Sun and NeXT, as well as a simple Unix encoding scheme. The user selects a reply format after recording; if the recipient is also found in the user's name and address database, the format is stored there to eliminate the need to ask again.

We did not invent yet another multimedia message format for mailtalk, but rather chose to operate with a number of existing ones. If the recipient uses one of the multimedia mail tool products, the message can be heard using the supplied graphical user interface; otherwise, the recipient invokes some operating system commands to convert the mail message to an audio file to play on the local audio hardware. Mailtalk also detects voice messages sent with these formats, processes them and plays the voice when the user wishes to read the incoming message.

This section has discussed the importance of speech technologies for remote access of information over the telephone. Although a portable computer and a modem are more useful in many circumstances, the advantage of voice-only access is that any telephone, be it in an airport, carried in one's pocket, or by the side of the road can be used as a terminal. This goes beyond existing voice mail or information retrieval services in several ways, the foremost being the variety of databases that can be kept just a phone call away. Telephone access to the workstation increases the utility of stored voice as a data type, as it is the logical medium for entering some kinds of information. Finally, telephone access to a family of applications allows integration and interoperability between applications, much as window systems already allow users to move text between windows.

VOICE IN PORTABLE COMPUTERS

The final environment for application of speech technologies is very small highly portable computers. Current portable technology supports several classes of computers. The "laptops" weigh two to eight pounds

or more, and have high-quality displays with full keyboards. Still smaller are the "palmtops," with mediocre displays and keyboards too small for touch typing. Because the keyboard, and to a lesser extent the display, limit further size reductions, voice could be used as the dominant medium of still smaller hand-held computers. A current generation of integrated circuits designed for the consumer digital answering machine market makes such devices very feasible in the near term.

How would voice be used in a hand-held? The primary benefit of such a computer would be its very small size, small enough to carry in one's pocket and hence always near at hand. This would make it ideal for recording quick notes or reminders. In an experiment based on a prototype using an analog tape recorder, Degen, Mander, and Salomon (1992) verified that the ability to mark and later identify and retrieve segments of recorded speech addressed user concerns of control and access to voice as a data type. Although encouraging, this study reveals many of the difficulties involved with navigating among voice segments with no display, and suggests that there will be a tight bond between portable voice devices and desktop computers with displays.

Eliminating the keyboard entirely allows a dramatic reduction in computer size, perhaps to something the size of a microcassette recorder, yet some means of data entry must be provided. Although large vocabulary recognition would be impractical in such a small device, limited vocabulary recognition coupled with digital voice storage could provide easy access to applications. The user might utter "Calendar . . . add . . . tomorrow . . . remember to send in the book manuscript." If the computer could recognize the first three words, spoken in isolation, it could record the rest and update the appropriate database.

Voice could be captured in a less structured form for taking notes, keeping a list of things to do, or outlining ideas for a project proposal or paper. A current project at the Media Laboratory uses speech recognition to select a directory (or *folder*) by name, and then allows voice notes to be recorded into these different directories. Interfaces based on speech recognition as well as buttons allow the user to scan lists, delete items, and move items between lists. The hand-held might be used primarily as a data capture device; the digitized sounds would be uploaded to the host computer on the desk where they could be manipulated and moved to the appropriate application using graphical interfaces. For example, Figure 5 shows the graphical user interface to a voice and text "things to do" list. In addition to recording entries over the telephone, as is done at present, voice notes could also be uploaded from the hand-held. Sometimes the text would be transcribed by the user; at other times it may suffice simply to be able to play back the speech as a reminder of the task or topic.

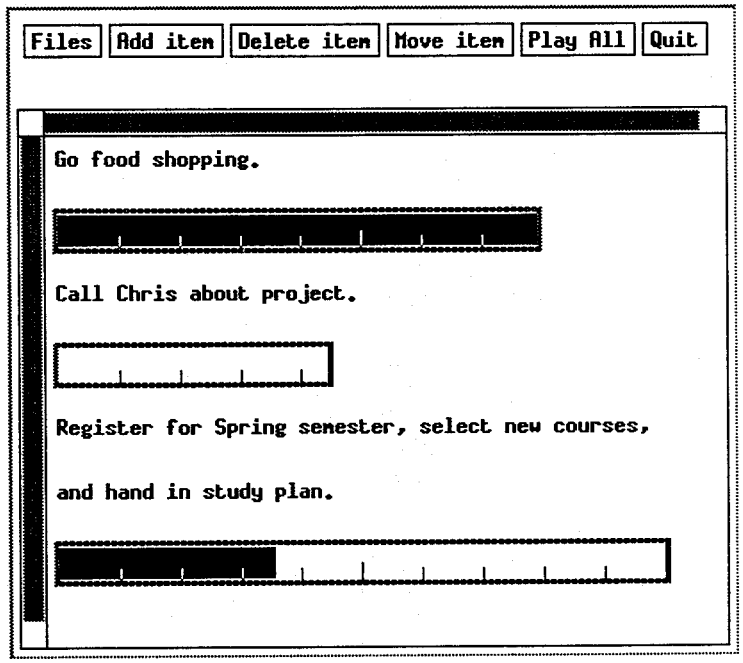


Figure 5. The visual user interface to a voice and text "things to do" application.

This suggested use of voice is not meant to claim that speech is the only alternative form of input to a hand-held computer. A current generation of portable devices support handwriting recognition and stylus-based user interfaces. Unfortunately, character input is slow and limited to printing well-formed letters; but there are situations, such as taking notes in a lecture, where slow character input is more viable than voice recording. Another technique for character input is the chorded keyboard, where a single hand is used to "type" by pressing multiple keys simultaneously, similar in concept to the steno machines used to transcribe courtroom proceedings. A marked advantage of voice recording is that it does not require visual attention; one could use such a device while walking, driving, or commuting on a bicycle.

Additionally, voice recording allows spontaneous capture of speech in our ordinary work situations. A digital voice recorder might be used to capture key portions of a meeting, or to record directions or instructions as they are spoken, without interruption. Or the user might wish

to recite portions, such as steps in driving directions, for later retrieval as a sequential list. Of course, as audio recordings become lengthier and less structured, it will become more important to provide navigational cues to the user along with acoustical means of skipping or scanning long or numerous voice passages.

These ideas are just the beginning. Because this is such a new venue for voice, its potential far outweighs our limited conceptions about the role of voice interfaces in conventional computers.

PUTTING IT ALL TOGETHER

This chapter has both extolled speech as an interface medium and lamented its weaknesses, as inherent in the channel and as artifacts of the capabilities of current technologies. The limitations of voice have to date relegated its deployment to specialized niche applications. I have suggested that careful matching of technologies to applications, along with attention to crafting user interfaces, can lead to successful applications of voice processing in our daily lives.

Key aspects of voice are its portability and our ability to use it as at a distance; an important liability is that it generally is not as effective as a keyboard for entering data. As a consequence, I have examined the role of voice in three environments for user interaction: in the office, over the telephone, and with a hand-held voice computer. Perhaps the most exciting aspects of speech interaction will be those that cross the boundaries of these styles of use. Voice on the desktop is much more enticing when it is coupled with remote telephone access or derived from notes taken on a hand held computer.

This synergy across the three environments will be the key to making voice an essential aspect of our computing environment. While a user may have little motivation to record voice into a calendar when a keyboard is available, recording may be the best means of entering data over the telephone or using a portable computer. Users of electronic mail may prefer text to voice, but voice comes into its own when replying to messages over the telephone. But applications such as these are much more likely to succeed when an associated graphical user interface can smoothly integrate voice and text media at the workstation. And although a hand-held computer may be ideal to capture ideas and project notes, a graphical interface to these recorded snippets can better allow a user to organize and navigate among them.

No single application of voice in computers seems destined to revolutionize the office, but voice employed across the range of applications and as a medium for accessing computers promises a rich and varied

new means of managing information. At least some of the technological components are in place for this vision to become a reality. The challenge is to develop creative applications and effective user interfaces to take full advantage of the power of speech.

REFERENCES

Ades, S., & Swinehart, D. C. (1986). *Voice annotation and editing in a workstation environment* (Tech. Rep. CSL-86-3). Palo Alto, CA: Xerox Palo Alto Research Center.

Allen, R. B. (1983). Composition and editing of spoken letters. *International Journal of Man/Machine Studies, 19,* 181–193.

Allport, D. A., Antonis, B., & Reynolds, P. (1972). On the division of attention: A disproof of the single channel hypothesis. *Quarterly Journal of Experimental Psychology, 24,* 225–235.

Angebranndt, S., Hyde, R. L., Luong, D. H., Siravara, N., & Schmandt, C. (1991). Integrating audio and telephony in a distributed workstation environment. *Proceedings of the Summer 1991 USENIX Conference,* pp. 419–435.

Arons, B. (1991). Hyperspeech: Navigating in speech-only hypermedia. *Proceedings of Hypertext '91* (pp. 133–146). New York: ACM.

Card, S., & Henderson, A., Jr. (1987). A multiple, virtual-workspace interface to support user task switching. *Proceedings of Human Factors in Computing Systems and Graphics Interface* (pp. 53–59) New York: ACM.

Chalfonte, B. L., Fish, R. S., & Kraut, R. E. (1991). Expressive richness: A comparison of speech and text as media for revision. *Proceedings of the Conference on Computer Human Interaction* (pp. 21–26). New York: ACM.

Chapanis, A. (1975). Interactive human communication. *Scientific American, 232,* 36–42.

Davis, J. R. (1991). Let your fingers do the spelling: Implicit disambiguation of words spelled with the telephone keypad. *Journal of The American Voice I/O Society, 9,* 57–66.

Degen, L., Mander, R., & Salomon, G. (1992). Working with audio: Integrating personal tape recorders and desktop computers. CHI'92 Proceedings. New York: ACM.

Englebeck, G., & Roberts, T. L. (1989). *The effects of several voice-menu characteristics on menu-selection performance* (Tech. Rep.). Boulder, CO: US West Advanced Technologies.

Gaver, W. W. (1986). Auditory icons: Using sound in computer interfaces. *Human Computer Interaction, 2*(2), 168–177.

Gaver, W. W., Smith, R. B., & O'Shea, T. (1991). Effective sounds in complex systems: The ARKola simulation. *Proceedings of the Conference on Computer Human Interaction* (pp. 85–90). New York: ACM.

Gould, J. D. (1983). Composing letters with a simulated listening typewriter. *Communications of the ACM, 26*(4), 295–308.

Gould, J. D., & Boies, S. J. (1978). How authors think about their writing, dictating, and speaking. *Human Factors, 20*(4), 495–505.

Gould, J. D., & Boies, S. J. (1984). Speech filing: An office system for principals. *IBM Systems Journal, 23*(1), 65–81.

Gould, J. D. (1982). Writing and speaking letters and messages. *International Journal of Man/Machine Studies, 16*(2), 147–171.

Hayes, P. J., & Reddy, R. (1983). Steps towards graceful interaction in spoken and written man-machine communication. *International Journal of Man/Machine Systems, 19*, 231–284.

Hindus, D. (1992). *Semi-structured capture and display of telephone conversations.* master's thesis, MIT.

Jelinek, F. (1985). The development of an experimental discrete dictation recognizer. *Proceedings of the IEEE, 73*, 1616–1624.

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

Kraut, R. E., Lewis, S. H., & Swezey, L. W. (1982). Listener responsiveness and the coordination of conversation. *Journal of Personality and Social Psychology, 41*(4), 718–731.

Lamming, M., & Newman, W. (1991). *Activity-based information retrieval: Technology in support of human memory* (Tech. Rep. 91-03). Cambridge, England: Rank Xerox EuroPARC.

Lee, K-F., & Hon, H-W. (1988, April). *Large-vocabulary speaker-independent continuous speech recognition using HMM.* Paper presented at the International Conference on Acoustics, Speech, and Signal Processing. New York.

Luce, P. A., Feustel, T. C., & Pisoni, D. B. (1983). Capacity demands in short-term memory for synthetic and natural speech. *Human Factors, 25*(1), 17–32.

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

Martin, G. L. (1989). The utility of speech input in user-computer interfaces. *International Journal of Man/Machine Studies, 30*, 355–375.

McPeters, D. L., & Tharp, A. L. (1984). The influence of rule-generated stress on computer-synthesized speech. *International Journal of Man/Machine Studies, 20*, 215–226.

Minneman, S. L., & Bly, S. A. (1991). Managing a trois: a study of a multi-user drawing tool in distributed design work. *Proceedings of the Conference on Computer Human Interaction* (pp. 217–224). New York: ACM.

Muller, M. J., & Daniel, J. E. (1990). Toward a definition of voice documents. *Proceedings of the 1990 Conference on Office Information Systems.* New York: ACM.

Nicholson, R. T. (1985). Usage patterns in an integrated voice and data communications system. *ACM Transactions on Office Information Systems, 3*, 307–314.

Ochsman, R. B., & Chapanis, A. (1974). The effects of 10 communication modes on the behavior of teams during co-operative problem-solving. *International Journal of Man/Machine Studies, 6*, 579–619.

Pisoni, D. B., Nusbaum, H. C., & Greene, B. G. (1985). Perception of synthetic speech generated by rule. *Proceedings of the IEEE, 73*(11), 1665–1676.

Rabiner, L. R., & Schafer, R. W. (1978). *Digital processing of speech signals.* Englewood Cliffs, NJ: Prentice-Hall.

Resnick, P., & Virzi, R. A. (1992). Skip and scan: Cleaning up telephone interfaces. CHI'92 proceedings. New York: ACM.

Rudnicky, A. I., Lunati, J. M., & Franz, A. M. (1991). Spoken language recognition in an office management domain. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* (pp. 829–832). New York.

Rutter, D. R. (1987). *Communicating by telephone.* New York: Pergamon Press.

Schmandt, C. (1984, August–September). Speech synthesis gives voiced access to an electronic mail system. *Speech Technology,* pp. 66–68.

Schmandt, C., Arons, B., & Simmons, C. (1985). Voice interaction in an integrated office and telecommunications environment. *Proceedings of the 1985 Conference* (pp. 51–57). American Voice I/O Society. AVIOS: San Jose, CA

Schmandt, C., & McKenna, M. (1988). An audio and telephone server for multimedia workstations. *Proceedings of the 2nd IEEE Conference on Computer Workstations* (pp. 150–159). New York: IEEE.

Schmandt, C. (1990). Caltalk: A multi-media calendar. *Proceedings of the 1990 Conference* (pp. 71–75). American Voice I/O Society. San Jose, CA

Schmandt, C., Ackerman, M. S., & Hindus, D. (1990). Augmenting a window system with speech input. *IEEE Computer, 23*(8), 50–56.

Schmandt, C., & Arons, B. (1986). A robust parser and dialog generator for a conversational office system. *Proceedings of the 1986 Conference* (pp. 355–365). San Jose, CA: American Voice I/O Society.

Schmandt, C., & Arons, B. (1989). Getting the word. *UNIX Review, 7*(10), 54–62.

Schmandt, C., Hindus, D., Ackerman, M., & Manandhar, S. (1990). Observations on using speech input for window navigation. *Proceedings of the IFIP TC 13 Third International Conference on Human-Computer Interaction,* pp. 787–793.

Spiegel, M. F. (1985). Pronouncing surnames automatically. *Proceedings of the 1985 Conference.* San Jose, CA: American Voice I/O Society.

Stifelman, L. J. (1991). Not just another voice mail system. *Proceedings of the 1991 Conference,* (pp. 21–26). San Jose, CA: American Voice I/O Society.

Wickens, C. D., Mountford, S. J., & Schreiner, W. (1981). Multiple resources, task-hemispheric integrity, and individual differences in time-sharing. *Human Factors, 23*, 211–230.

Zellweger, P. T. (1989). Scripted documents: a hypermedia path mechanism. *Proceedings of Hypertext '89* (pp. 1–14). New York: ACM.