MAS.632 Speech Interfaces and Mobile Devices
Fall 2008

6

Interactive Voice Response

This chapter is about building *interactive* computer systems that use speech, either recorded or synthesized, as their sole form of output. The focus will be on the advantages and liabilities of speech as an output medium and on interaction techniques for applications to employ speech output most effectively. It is essential to understand the limitations of voice output, both when deciding whether speech output is appropriate to an application as well as when designing details of the interface.

Chapter 3 discussed techniques for digitally encoding and compressing stored voice, and Chapter 5 described the process of converting text to speech. This chapter considers attributes of both digitized and synthesized speech as components of interactive systems. Although most of the growing number of commercial voice response systems employ recorded speech, this chapter also includes applications of synthesized speech, which are emphasized in the case studies.

Chapter 4 discussed the role of stored voice as a data type, primarily in the context of its role as a compbnent of an electronic document. For most of the examples in that chapter, a visual user interface was employed because most documents contain visual material. In this chapter we consider applications with no visual interface by focusing on telephone-based interactive voice response systems. Such systems are interactive because the caller employs touch tones, or occasionally speech recognition, to make selections, and the application responds appropriately using speech output.

For the purposes of this chapter it is assumed that user input is accomplished with touch tones as the basics of speech recognition are not discussed until Chapter 7. Touch tones provide an abstract user input, that has no innate meaning

except in the context of an application. In contrast to speech recognition, touch tones are unlikely to be detected incorrectly by application hardware; this allows some simplification in discussions of user interaction techniques. This chapter ignores the technical details of the operation of telephones and transmission of touch tones except as they impact interaction techniques; details of the operation of telephones and telephone networks is included in Chapter 10.

This chapter also contrasts the assets and liabilities of speech as an output medium. After this, the focus shifts to details of interaction techniques for voice output and then to issues in menu selection or data entry for telephone-based voice systems. The final section offers three case studies as sample applications illustrating the techniques described in this chapter.

LIMITATIONS OF SPEECH OUTPUT

Chapter 3 raised the issue of intelligibility of recorded speech, mostly as an issue of speech coder design or selection. In the discussion of speech synthesis in Chapter 5, intelligibility was again highlighted as a major item of concern. Because synthetic speech is generally of poorer quality than encoded human speech, sources of pronunciation error and effects of listener learning are important when using speech synthesis.

Although intelligibility is certainly one factor in determining the appropriateness of using voice output in a particular environment, there are a number of other issues that also need to be considered in designing a user interface that uses any form of speech output. Speech is both slow and serial as well as "bulky," requiring listener attention to scan the contents of recorded voice files. Speech broadcasts within an acoustic environment, which raises issues of personal privacy and possibly disturbs one's neighbors. These characteristics are not limited to synthetic speech although they may be exacerbated by its decreased intelligibility and the suspected increase in cognitive load on the listener. These issues, as much as intelligibility, place constraints on whatever interaction techniques might be employed by an application.

Speed

Speech is slow; listening is much slower than reading. Typical speaking rates are in the range of 175 to 225 words per minute. People can easily read 350 to 500 words per minute. A 9600 baud computer terminal can receive nearly 10,000 words per minute, while a 1200 baud modem (nearly obsolete in current communications technology) transmits more than 1000 words per minute. Both recorded and synthesized speech can be sped up, but time-scaling to more than 1.5 to 2.0 times the original speed reaches an upper limit and requires considerable attention by the listener. Reading rates can be increased with practice as well.

As a consequence of its slow speed, using voice to access information can become quite tedious. A few paragraphs of text may take a minute or more to speak. The information to be spoken, either by synthesis or digital playback,

needs to be concise and direct. One of the banes of voice mail systems are loquacious users who leave long, rambling messages. Because of speed considerations some text-based interactive computer activities such as browsing through network news may be unsuitable for speech output.

In comparing the speed of speech to conventional screen and keyboard interaction, it is important to note that the speed difference is not symmetrical with respect to input and output. Although listening is slower than reading, speaking is faster than writing or typing in many situations. For example, the author of a recorded voice message has a speed advantage over the listener; this may likely influence the circumstances under which voice may be chosen over text if both media are available.

Temporal Nature

Speech is an acoustic phenomenon consisting of variations in air pressure over time. By definition, once spoken the speech is gone. If the listener was inattentive or absent when a message was spoken the information is lost. This is the antithesis of graphical user interfaces with dialog boxes that persist until the user clicks on a mouse button. The temporal nature of speech requires a means of focusing the listener's attention to attend to sporadic auditory events; many international airports play a distinctive chime before speaking a boarding announcement over the public address system.

Serial Nature

A stream of voice conveys only one word at a time;[1] we must wait for the first word to be spoken before we can hear the second. If a menu is displayed on a terminal, all the items appear very quickly and the user then visually scans them until a choice is made. With speech each item must be recited in turn; once spoken it is gone unless the user remembers it. If the user misses a portion of the message, it must be repeated in its entirety. Because this serial nature of speech places additional memory requirements on the user, menu items and other application responses must be short and precise.

A consequence of the temporal nature of speech and the serial presentation of menu items is the interaction between presentation sequence and the timing of user input. As shown in Figure 6.1, it may be difficult to detect which choice a user responds to if expected to make a selection immediately after hearing the desired choice.

Bulkiness

The storage space required for recorded voice was of much greater concern five or ten years ago than today. As the costs of computer memory and magnetic disks

[1]This is not quite true. Speech is not just a string of disjoint phonemes; there is a certain amount of parallel transmission of phonemes in that the allophonic and acoustic realization of a phoneme is influenced by its neighbors.

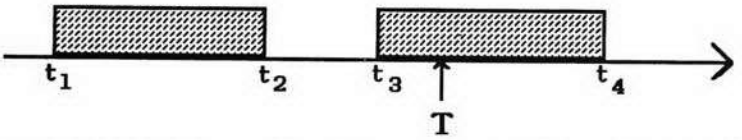

Figure 6.1. Difficulties synchronizing user input with speech output. Two utterances are spoken spanning t_1 to t_2 and t_3 to t_4. At time T the user responds. Is this response a delayed reaction to the first utterance or a very quick response to interrupt the second?

have dropped, storage has become a less important issue. However, managing stored voice and providing the user with sufficient access to it remains a difficult problem for other reasons.

If we wish to hear information recorded in a sound file, how can we determine which file to play, or what part of the file to play if it is many seconds or minutes long? Retrieval can be facilitated if a text-based description accompanies the sound; this can be stored as "header" information in the sound file itself. Situational cues, such as when the recording was made, the identity of the speaker, and which computer application created the recording may also help limit the number of files that must be searched. But such information is of marginal additional value and in no way approaches the power of keyword searching among text files.

Finding a small number of words in a large quantity of speech is known as keyword spotting; unfortunately, this is one of the most difficult forms of speech recognition and currently not widely available. Instead, a person must listen to the recorded speech to find a particular piece of information. Listening requires *attention*. Although the user can perform other tasks while executing a long text-based keyword search, this is not possible while listening.[2] Although both synthesized and digitized speech may be sped up during presentation, this demands still more attention from the listener.

Privacy

Voice announcements, messages, or warnings are much more public than visual notification. A message displayed in a small font on a computer screen may be visible only to the typist or someone sitting nearby, but voice messages may be heard across a room. In addition, if a visitor to an office tries to read the screen, this is apparent to others by their gaze; eyes are more directional than ears. Excessive use of voice output can lead to a distracting and tiring work environment, and sensitive messages should never be voiced without a guarantee of privacy. But this characteristic of speech can also be advantageous in some classes of applications; speech is useful in public address systems precisely because

[2]We *can* perform other visual or manual tasks while listening but still must devote significant attention to the auditory channel.

everyone can hear the announcements regardless of the focus of their visual attention. Voice can be similarly used to alert the computer user in an office when the user requires mobility or is not paying attention to the screen.

ADVANTAGES OF VOICE

Given these significant liabilities of voice, what advantages could possibly justify its use? Chapter 4 suggested several roles of voice in conjunction with text in the context of document-oriented applications. Voice is richer than text and more expressive of information that may be tentative or subject to multiple interpretations. Voice is useful for annotation because it acts as an added dimension for placing comments on an otherwise two-dimensional view of a document page. The rest of this section describes assets of voice beyond document-oriented uses.

For many styles of application discussed in this chapter, the main advantage of speech is its **accessibility** via telephone-based interfaces. Telephone-based applications can be used from nearly any telephone. Most public telephones in the U.S. and Canada are equipped with touch tone keypads, but this is not true in some other parts of the world, notably some European countries. When touch tones are not available, speech recognition must be employed; unfortunately, speech recognition over the telephone is much more error prone than touch tone decoding.

The accessibility advantage of voice also applies to the authoring or creation of information. The ability to capture speech as a recording for subsequent playback allows data to be entered into a system when no other means of input is available, such as recording notes on a pocket tape recorder, dictation, or leaving messages to oneself on one's answering machine. Telephones allow callers to record information at a distance; for example, Resnick describes a community bulletin board enabling callers to browse as well as enter events in a public events database by recording them [Resnick 1992a]. Voice mail allows subscribers to record messages, and Chapter 12 describes telephone access to desktop computer applications that allow a user to record voice entries in a personal calendar. Telephone-based recording also can be used for functions such as order entry by filling out an audio "form" for later transcription.

Although the previous section describes the **broadcast nature** of speech as a liability, in other circumstances it is an asset. The ability to reach a number of listeners simultaneously over a broad area, such as airport flight departure announcements, simply could not be provided by visual displays.[3] Because our ears are much less directional than our eyes, auditory cues can be delivered with more confidence that the intended audience will be more able to be attentive at the moment of presentation. Voice output is effective when the user is mobile or

[3]It should be noted that displays clearly are of value. Imagine having to locate your departure gate by listening to a continuous recitation of all flights departing within the next hour!

less likely to be focusing on a display. The user may be staring out the window, reading a book, or walking around the office and not notice a new message on the computer screen.

Voice can also be useful even when the intended recipient is in a more restrained physical environment such as sitting in an office using a workstation or in an airplane cockpit, because it provides an alternate channel of interaction when the user's **hands and eyes are busy** elsewhere. If appropriate and informative words are spoken such as "System shutdown in ten minutes" or "New mail from Kaya," then users would not need to shift their gaze to the screen.[4]

Spoken messages offer a performance improvement in addition to providing the ability to audibly distinguish the message content. Experimental evidence (see [Allport *et al.* 1972, Wickens 1981, Treisman and Davies 1973]) indicates that if a user is performing multiple tasks, performance improves if those tasks can be managed over independent input/output channels, or modalities. For example, although we cannot listen to two conversations simultaneously, subjects were able to sight read piano music while repeating what they heard with no decrease in proficiency [Allport *et al.* 1972]. How do such experimental results apply to voice in an office environment? Voice notification to indicate the status of a task to which the user is not paying immediate attention or for which the associated window is not visible could effectively deliver a message with minimal disruption of the visual attention a user is simultaneously applying to other tasks.

Speech synthesis affords systems the capability of **media translation** to turn text documents into voice. Although this seems to be a trivial continuation of the earlier discussion about remote access, it should not be dismissed quite so quickly. Much of the current work in multimedia computing uses different media for distinct purposes to present disjoint, although related, information. For example, a voice narration accompanying the display of a simulation program is meant to comment on the program's graphics rather than act as a substitute output medium. Speech synthesis lets text data be converted into voice for any purpose. Some blind users make remarkable use of it. Voice output is a first step towards employing multimedia technologies to make computers accessible to a variety of users under many different circumstances.

DESIGN CONSIDERATIONS

An understanding of the limitations of speech as an output channel enables us to design better voice interfaces. Speech can be used effectively for computer output, but care must be taken to avoid allowing the liabilities of the medium to overcome the usefulness of the application. The next two sections of this chapter

[4]Displaying the text message *in addition* to speaking it is necessary for several of these applications. If I am out of the office when an important notice arrives, I will miss the speech output but may look at the screen when I return.

outline a number of design considerations and discuss some interactive techniques in this light.

Most of the limitations of voice output described earlier can be summarized by the broader observation that speech takes place *over a period of time.* The amount of time required to listen to a voice recording, the necessity of reciting menu items in sequence, and much of the difficulty associated with the bulky nature of voice are all a direct consequence of the temporal nature of speech.

Speed is thus one of the primary factors distinguishing voice from other interface media; for example, most graphics applications can be written with the assumption that display operations such as drawing lines or text are performed nearly instantaneously. Much of the time the application is idle while the user reads text, picks a menu item, etc.; this is punctuated by sporadic user input and display update. For telephone-based interfaces, the exact opposite is true; presentation accounts for most of the total time the user interacts with the application.

Voice response systems must therefore strive to minimize the amount of time required to retrieve information: they must be brief, selective about content, and interruptible. Applications must plan what to say so as to avoid utterances that are irrelevant by the time they are spoken. A real-time speech output system may have to formulate conflicting goals and choose among them if there is not sufficient time to say everything, which requires estimating how long it will take to speak an utterance. The awareness that speech takes time (and acceptance that time is a commodity of which we never have enough) should permeate the design process for building any speech application. This argues in favor of shortening spoken menus and precludes many applications that may be quite feasible in text such as browsing through a newspaper. Time motivates strategies to filter an application's output requiring careful judgment as to the relevance of a specific system comment. The remainder of this section explores these design considerations in more detail.

Application Appropriateness

Because speech is so difficult to employ effectively, discretion is essential in determining effective voice applications and in choosing the appropriate speech output device. Poor quality speech, whether synthesized or recorded, quickly becomes annoying unless it serves a clear purpose. For example, speech output has been employed effectively as a warning medium in airplane cockpits. On the contrary, few car drivers appreciate a tinny voice reminding them to close the doors of their automobile. The cockpit is a busier place, and the consequences of an error may be more severe. A more effective use of speech in the car may be traffic information (drivers already tune to commuter traffic reports on their radios) or as part of a navigation system (as described later in this chapter).

In retrospect, another example of a misguided speech interface is the recitation of a customer's purchases at a cash register. As each item is scanned by the bar code reader, its description and price are announced. The original motivation for

these systems was a concern over customer skepticism at the accuracy of the bar code readers. However, public acceptance of this technology came rapidly, and as a result the voice output soon was considered annoying, both to the cashier who had to listen to this drivel all day as well as to the customer who perhaps was not interested in having his or her purchases announced to the surrounding patrons. A text version of this information, both on a register display as well as on a detailed receipt, adequately satisfies customer doubts.

Speech offers more opportunities in telephone-based interfaces because it offers a means of remote access to important information. Such systems are the focus of much of the remainder of this chapter. However, for such applications to be useful, access to the information being requested over the phone must be a priority to the user. Under most circumstances, few sighted computer users would choose to hear their electronic mail synthesized when they could read it. But timely access to certain messages may be important enough to recipients to make telephone access to messages a valuable application.

Speech quality interacts with the perceived utility of an application. If both can be used equally effectively, it is rarely the case that synthetic speech is more appropriate to an application than recorded speech. If only a few spoken messages are needed, prerecorded speech generally is adequate, less expensive, and likely to be better understood. For some data, however, synthesis is essential.

Data Appropriateness

Since speech is slow, only essential data should be spoken. For example, while speaking electronic mail, such information as dates, times, and many other extraneous fields detailing message routing should be discarded. The point is less "Is this data useful or interesting?" than "Is this data *really* necessary?" Removing extraneous information can save the listener a substantial amount of time and increase the likelihood of acceptance of the application. Information that is spoken should be presented in a simple and accommodating form. Computer time stamps often include the date and time in seconds or even milliseconds, but a person likely prefers to hear "this morning at 10:30" or "yesterday evening" instead of "Wednesday October two 1991 at seven twenty-one and twenty-three seconds."

The wording of recorded speech output is also critical. Prompts should be as brief as is practical; excessively friendly or chatty systems wear thin. Some voice mail systems offer their users the option to choose between normal and abbreviated prompts that allow experienced users a more streamlined interaction. Whether a system is oriented towards novice or experienced users should depend on the expected user population. A service such as national weather information may have a broad base of occasional users, but a service to report the availability of a particular computer system may experience frequent use by a small number of individuals who benefit from terse phrasing.

If the data can be spoken as many variations in a sequence of information elements such as the set of 10 digits spoken as components of a telephone num-

ber, recorded speech is superior to synthesized voice. An example of such an application is telephone presentation of weather information; a forecast could be composed from small units such as "The weather in . . . Boston . . . is partly cloudy . . . with a temperature of . . . seventy-eight degrees."

Because listener comprehension of synthetic speech improves rapidly with exposure, frequent users of specialized applications may find it acceptable regardless of the data. If the data is human-authored, e.g., electronic mail or entries in one's calendar, synthetic speech is the only practical means of speaking. The data may be difficult to synthesize well, however. Surnames are particularly hard to pronounce correctly as they are derived from many languages. Complete sentences such as those found in electronic mail are more easily comprehended than isolated words or phrases that might be found in a meeting agenda announcement.

For all these reasons, the data must drive presentation strategies. Filtering, terseness, and use of an appropriate form of voice output are all essential.

Responsiveness

User input should trigger some voice response, either an acknowledgment of the input or actually speaking the information the user just requested. But because speech is slow, explicitly echoing every keystroke is not appropriate. If an application offers frequent feedback, then the user can be confident that input was detected; application silence might cause the user to press a key repeatedly and lead to errors.

If a system employs multiple menus, feedback can be a form of navigational cue by which each menu identifies itself ("top level," "main menu," "configuration menu") before presenting its choices. Such a cue would also be appropriate if the user enters ill-formed input; the system can remind the user what it thinks he or she is supposed to be doing. ("You are entering a telephone number. Please enter the number terminated by a pound sign.") Similar vocal prods can also be made during the invocation of time-outs such as when the user does not make a selection after hearing an entire menu of choices.

Speech Rate

One method of increasing an application's apparent responsiveness and minimizing message retrieval time is to speed up voice output. Most speech synthesizers support commands to vary speech rate, expressed as words per minute. Informal observations of experienced users accessing electronic mail over the telephone at the Media Lab indicate acceptable comprehension at 300 to 350 words per minute with a synthesizer with a default speech rate of 180 words per minute. Blind computer users can understand significantly faster synthesized speech. Recorded speech can be sped up for playback as well as described in Chapter 3.

Time compression of speech output is most effective for experienced users who are already familiar with the application, especially its menus and interaction structure. The more predictable the output, the faster it can be spoken.

Interruption

A voice interface should almost always be interruptible. This is a direct consequence of the slow transmission time for a message. A passage may be irrelevant, unintelligible or redundant, and the user should be able to skip over it. If the system can detect input while speaking by allowing "type ahead" control (most commonly controlled by touch tones), an experienced user need not endure a long-winded introduction or menu repetition. This allows a clean compromise between abbreviated instructions for expert users and lengthier tutorials for novices. Unfortunately, users tend to be more hesitant to interrupt than system designers might expect or desire. It may be that users lack confidence in their ability to recollect the proper keystrokes or simply cannot be bothered to learn them, so it is helpful to keep explanations short.

One problem with supporting interruption is that it may be difficult for the system to determine exactly where the interruption is intended; this was alluded to earlier in this chapter. There is a time delay of at least several hundred milliseconds between the transmission of text to the synthesizer and the onset of speech output, and an additional delay while the user makes a decision; this is a challenge when building an interface that presents choices as "When you hear the item in which you are interested, press any key." Even the slightest of delays due to the user's processing of the information may result in an erroneous interpretation as the next prompt begins to play.

Another problem with interruption is that it may not be possible in a telephone-based system utilizing voice recognition for user input. Analog telephones are two-wire systems; the voice from each party is mixed in the circuit. This implies that the speech recognizer hears the system's transmitted speech as well the user's voice, and this prevents reliable recognition because the recognizer is not able to discriminate between speech sources. Touch tone recognition is not as difficult as it is acoustically dissimilar from speech.

A solution to this mixed channel problem is available to allow the user to "talk over" the system's speech output to the speech recognizer. Since we know which signal corresponds to the spoken speech output, we can "look through" the signal by using it as a reference and sampling the telephone line during short silent periods in the system's speech. These silent periods may correspond to some of the longer stop consonants or pauses that occur between phrases. If a significant signal is detected by the computer at these times, it must be the user trying to interrupt; speech output can be stopped so recognition processing can be begun. Such a configuration, shown in Figure 6.2, was built and proved successful in limited testing in the author's laboratory.

Repetition

A means must be provided for the system to repeat itself. This applies both to reciting menus several times as well as to repeating sentences of human-authored text or computer-generated discourse. Repetition will likely be combined with an interruption strategy if long messages are being recited.

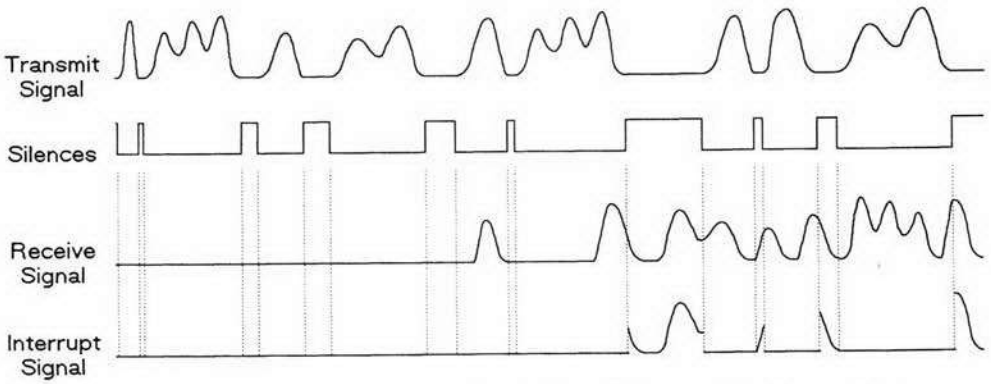

Figure 6.2. A configuration to "look through" transmitted speech to detect user interruptions.

How should a repetition be spoken? In the case of a menu, it suffices to repeat the menu items with identical wording as the items are often short and the listener most likely was just having trouble recalling them all. If output consists of recorded speech, the application can only replay the recording, possibly doing so at a slower speed. If the text is a passage of computer-generated discourse, it may be effective to generate an alternate phrasing of the text in the hopes that this would help with any pronunciation errors. However, this is also problematic as the user may be expecting a word-for-word repetition and thus be put off balance. There is little empirical evidence one way or the other.

In the case of synthesizing human-authored text, however, the problem could be either a spelling or typographical error, failure of the synthesizer's text-to-phoneme rules, or lack of user attention. One repetition strategy is to speak the same text again at a slower rate in the hope that this enhances intelligibility. The system to read electronic mail over the telephone, which will be described as a case study, chose to repeat more slowly and then to switch to "spell mode" and pronounce the text letter by letter. Spelling a mail message or almost any other data is tedious. An alternative strategy involved passing the text passage through a spelling checker program and during repetition combining a slower speech rate with letter-by-letter spelling of any words not found in the dictionary. This technique helps catch text-to-phoneme errors as proper names are likely to be pronounced poorly but would not be found in the dictionary. Most typographical errors would also fail the spelling check and thus would be spelled out.

Exception Pronunciation

Words may be pronounced incorrectly. Ideally, the programmer should be able to add to the morpheme dictionary in the synthesizer, but no synthesizer product currently provides this capability. Some synthesizers allow for alternative pronunciations of a small set of words to be downloaded into the synthesizer, but if the synthesizer only uses this list with simple string substitution, it is not a

very powerful capability. For example, if "Schmandt" is in the dictionary, will "Schmandt's" be pronounced correctly?

Because of their limited capacity and literal string translation (instead of direct application access to the morpheme pronunciation tables), synthesizer-based exception dictionaries are not very powerful. Improved functionality can be provided in more capable workstations that might employ speech synthesizers. In fact, many workstations are now able to perform real-time speech synthesis entirely with software.

Instead of using simple exception lexicons, applications can better benefit from domain-specific text-to-phoneme rules. Consider several examples. My phone number, spelled 253-5156, should not be pronounced "two hundred fifty three hyphen five thousand one hundred fifty six." If the string "253-5156" is known to be a telephone number, it can be pronounced properly as three digits, a pause, and four digits. Postal Zip Codes are similarly pronounced digit by digit but street addresses are pronounced as ordinary numbers. My email host, "media-lab.mit.edu," is pronounced "media hyphen lab dot m i t dot e d u" by those familiar with Unix and Internet addressing.

While it is difficult to employ this knowledge to speak free-form human-authored text, much of the data to be synthesized is highly structured. Electronic mail headers, name and address databases, telephone lists, and appointment calendars each contain much structured data, which can benefit highly from string processing before synthesis. In this way, applications can employ much more intelligible synthetic speech.

Multiple Voices

Many synthesizers have the ability to use one of several vocal tract models or "voices." Switching voices may be invoked by software that offers a selection between male, female, deep, or raspy speech. The use of several different voices *may* make it easier for the user to discern various separate system functions, such as a command from data or a narrative mode from a multiple-choice mode. Realistically, however, even though a synthesizer may support multiple voices, a primary voice generally emerges as being significantly more intelligible than the auxiliary voices. Rosson studied users' responses to synthetic voices while varying a number of characteristics of the vocal tract model; she discovered that users preferred different speech qualities for different tasks, which she categorized as information providing, entertainment, and feedback [Rosson and Cecala 1986].

Little work has been done to determine how effective such a technique is in practice although frequent users of synthetic speech often modify pronunciation. For example, a telephone pager at the Media Lab uses speech synthesis to announce calls. Each user of the system is able to configure his or her own personal announcement message, and some users have included in their message the necessary escape sequences to change the synthesizer's voice to better differentiate their call announcement from others. Other users achieve a similar effect by modifying the prosody of their message or the duration of some of its syllables.

USER INPUT WITH TOUCHTONES

This chapter emphasizes telephone-based applications of speech output. These applications use touch tones for user input; when the user presses a button on the telephone keypad, a tone is generated that is decoded into one of 12 symbols at the application end, usually using special but inexpensive hardware. This section discusses a variety of user interaction techniques employing touch tone input.

User input can be divided into two classes: making a selection and entering data. Menus are a selection mechanism, and the response to a yes-or-no question can be thought of as a menu limited to two choices. Data entry may be numeric using the obvious numbers on the telephone keys or may involve the spelling of words by making use of the letters that also appear on most keys. As already noted, telephones outside of North America may not have letters in addition to numbers.

When presenting menus, how should keys be described? Although most references are straightforward, a few are confusing. The "0" key is likely to be called "oh," both because that is how the digit is usually pronounced in phone numbers as well as because it is used to reach the operator. The "#" key is alternately called "pound" (as in weight), "sharp" (from music notation), or "number sign"; in the telecommunications industry it is very rarely referred to as the "octothorpe." The key labeled "*" may be called "star" or "asterisk."

Menus

Menus may be either **temporal** or, more commonly, **enumerated.** In an enumerated menu each choice is presented accompanied by a number: "Press one to hear your messages, press two to send a message, press star if you are finished." A temporal menu asks the user to press a key when he or she hears the choice desired and pauses between each choice: "Hit any key when you hear the city you want. Somerville . . . Cambridge . . . Boston. . . ." A binary menu (i.e., yes or no) is simply a degenerate case and may be implemented in either way. If a default is to be assumed, then a short form can be used such as "If you want to send a reply, press any key now."

Each style has both advantages and disadvantages. Timing is more critical with the temporal menu style because the application must know when the prompt was spoken and measure a consistent amount of time before presenting the next choice. In the event of a user entering input just at the beginning of the speaking of a choice, it may be unclear as to whether the user's response was an early acceptance in anticipation of the next choice or a delayed response to the previous choice. The enumerated menu also needs a timer to handle circumstances in which the user makes no response to the enumerated list; the usual approach is to repeat the menu after the predetermined time-out period elapses.

An advantage of the enumerated over the temporal menu is that the experienced user need not listen to the entire list but can interrupt its recitation by entering a choice. The temporal menu necessitates waiting until the desired

choice is presented.[5] But if the list of choices changes (perhaps it is a list of voice mail users and a new employee joins the organization), then the associated numbering may change and the experienced user who types ahead may make an incorrect selection. The enumerated menu also requires the user to keep track of both the choice and the number associated with the choice; with a temporal menu one can concentrate exclusively on the list items themselves.

An issue with all menus is their size. Human factors folklore suggests that four items is the comfortable number. In a system implementing voice mail for athletes at the 1984 Olympics, Gould *et al.* found four items to be too many based on evaluations of a prototype system [Gould *et al.* 1987]. In a somewhat more contrived test task based on a laboratory experimental setup, Englebeck and Roberts [Englebeck and Roberts 1989] confirmed the proposal of four items, but note that a larger number of items may be more effective for experienced users. In their study, two important factors influenced the choice of an acceptable menu size: interruptions and user confidence. Although users are normally hesitant to interrupt the system, as menu items increase users appear more likely to interrupt when their choice is heard. This interruption generally results in faster selection time, although it could indicate frustration at the long menu. Confidence in the choice about to be made is also key and suggests that the phrasing of menus is very important. When the user hears a choice that is clearly the desired selection, interruption is more likely.

With an enumerated menu another issue is whether it should be presented in the order of action . . . key ("Reply press one") or key . . . action ("Press one to reply"). According to the Englebeck and Roberts study, users were more likely to interrupt with the action/key presentation. Additionally, this order was preferred in the subjective evaluations of tests using the two options.

Data Entry

Under some circumstances user input is required and a list of menu choices is impractical simply because there are too many possible selections (such as looking a name up in a company telephone book). In this case, to gather user input the system must speak a prompt, receive the responding keypresses, and detect the termination of the input. User input may be **numeric** or **alphabetic.**

Numeric input has many uses and is relatively easy to explain to the user as we are all quite used to dialing numbers to place calls. As part of an interactive service, a user may be required to enter a credit card number, telephone number, or the digits of an address. Services offering weather reports for locations around the country may ask for the telephone area code for the location. Interactive banking services prompt for an account number or a dollar amount.

Alphabetic input is somewhat more awkward. Fortunately, in North America the telephone keypad has letters inscribed on the keys remaining from the days

[5]A temporal menu can eliminate the wait by allowing the user to press a special key to skip to the next item [Resnick and Virzi 1992].

when telephone exchanges were referred to by a name instead of number.[6] This alphabetic labeling tends to get carried over into other public keypad style devices such as automated bank teller machines. But there are two problems with this arrangement: each key maps into three letters (see Figure 6.3), and the letters "Q" and "Z" do not appear on any key of the telephone.

For the two missing letters several options are in use. Both letters may be mapped to the otherwise vacant 1 key; this seems the most common alternative. They are also sometimes mapped to the 0 key. For the implicit key-to-letter decoding, described below, they can be assumed to occupy the keys on which they would appear if they had been left in the alphabet. In this scheme, Q appears invisibly on the 7 (PRS) key and similarly Z appears on the 9 key.

Alphabetic input techniques may decode the three-to-one letter-to-key mapping either explicitly or implicitly. Explicit mapping requires the user to type two keys for each letter; the first key chooses a group of three letters and the second selects one from this set. The second key may be constrained to one of the set 1, 2, or 3 or better could be any key with the column position of the key selecting the first, second, or third letter. For example, the letter H might be selected by entering 4 5; 4 selects the GHI set, and 5 from the middle column selects the middle letter from the triplet.

The implicit letter mapping technique accepts the confusability of the user's input and attempts to decode the input by matching it against the set of valid inputs. For example, while selecting a name from a menu (see Figure 6.4), the user is really picking one of a set of names, each of which may have a distinct touch tone "spelling" uniquely specifying that each letter is not necessary. In fact, in the illustrated example, four of the names would be uniquely specified after the first touch tone. (One might, however, wish to gather all five digits to confirm that the user was not trying to select a name not on the list.)

Explicit spelling requires twice as many keystrokes but is required for some tasks, e.g., if a new subscriber to a service is asked to spell his or her last name to create a new account. In the case of spelling to make a selection from a list, implicit character mapping may be quite adequate, faster, and easier to explain to the user; examples include looking up a last name in an electronic database, spelling a street name, or logging in to a computer account.

A problem with implicit spelling is the degree of overlap between the items being selected and their keypad equivalent. For example, the login names "Leo" and "Ken" are both spelled "536." In such a case, the collision must be detected by the system and an alternate selection mechanism invoked, e.g., "If you mean Leo, press one; if you mean Ken, press two." (Note that this is a menu and could be either enumerated or temporal).

Clearly the degree of overlap between the items is a function of the particular set of items, and therefore the confusability of implicit spelling will vary from

[6]Where I grew up in Chicago we were in the **ED**gewater exchange, although this convention is no longer followed.

Figure 6.3. The labels on a telephone keypad.

database to database. But even explicit spelling can encounter collisions between choices. Although the login IDs mentioned above may be unique on a particular computer system, for other kinds of databases uniqueness is less likely. There may be many "Smiths" in a large company, and there are quite a few "Cambridge" streets in the Greater Boston area, for example. Whenever this occurs, an alternative selection will have to be provided regardless of the data input method.

Implicit spelling may be very effective. For many databases, confusability due to letter-to-key mapping is less significant than collisions due to duplicated spellings. For example, Davis [Davis 1991] reported on two databases of names of job applicants and students at a university. In one list of approximately 9,000 individuals, 40% of them shared a last name with at least one other individual, but only 8% of all names collided with another name using implicit spelling. A second list of 25,000 individuals exhibited 38% shared names with 21% of the names colliding.

Choices	Touch tone spelling
Brown	27696
Davis	32847
Green	47336
Jones	56637
Smith	76484
South	76884

Figure 6.4. A menu of last names and the associated touch tone spelling.

For either numeric or alphabetic input, termination of the input sequence may also be implicit or explicit. If the input is of a fixed length, e.g., the three digits of an area code, no terminator is necessary. If the input is of variable length, input may be terminated by a specific key, the system may detect that the user is finished by a pause, or the system may automatically complete the input as soon as a unique choice is identifiable.

Implicit termination requires a timer. The system asks a user for input and gathers digits until the user has paused for a predetermined period of time. The time-out may be longer during the period before the user has entered any digits to provide for thinking time. In the event of a time-out without any input, a help mechanism should be triggered. **Explicit termination** requires the use of a key that cannot be confused with user input. The "#" key is often used for this purpose.

Explicit termination requires more explanation to the user than does implicit termination and also requires that one remembers what to do at the end of input. Explicit termination also requires a timer as it must cope with the user who inputs the requested data but then forgets the terminator. Explicit termination is faster if the user does remember the terminator as the time-out period is not necessary. Explicit termination can also accept multiple, sequential fields of input more easily than implicit termination, such as the house number and street name of an address as the user can type each in sequence indicating segmentation of the items with the terminator key. For example, I could enter "E 1 5 * 3 2 7" as my building and room number without waiting to hear a prompt to enter my room number. To be more general, explicit termination allows type ahead; if the user knows what the system will ask next he or she can start to enter it without waiting for the time-out.

Because the input is of variable length, both input techniques should perform range checking on the input if possible (for example, the number of digits in a valid house number) and warn the user when errors are detected. It may be that the user forgot the delimiter and started to enter the next field.

Hybrid termination schemes are possible. One example is **optional termination;** if the user enters the termination key, then the input is complete; otherwise input is ended by a time-out. This approach allows the experienced user to save time, while not confusing the novice about the use of a terminator. Such a scheme is used for dialing the number for an international telephone call, since there is considerable variation in the length of international telephone numbers. The local telephone switch does not know how many digits to collect before connecting to the destination country and therefore must wait for the caller stop dialing, using a time-out. The caller can terminate the dialed digits with a "#" to indicate the end of input, avoiding the delay caused by the time-out.

Another hybrid approach allows automatic completion of user input when a unique choice is made; this saves keystrokes for the user. In the simplistic database illustrated in Figure 6.4, for example, the first digit will uniquely select one of the first four names but "Smith" and "South" require three digits for a unique choice. With automatic completion the system responds by echoing the selected choice as soon as it can be determined. But the user may be in the pro-

cess of entering the next digit(s) anyway so the system must quickly determine whether the additional input is a continuation of the spelling or the next step in the user interaction. One technique reported in [Gould *et al.* 1987] is to "absorb" this further input as long as it conforms to the remaining digits in the selected item.

Another shortcut for selecting alphabetic choices by spelling is user-initiated completion. A special key is used to jump from spelling input to a menu style selection. For example, to continue with the name database example, the user could type "3*" and get "Davis," which is unique. But entering "7*" or even "76*" would invoke a menu to select between "Smith" and "South." With user-initiated completion, the user gets to choose between pressing more digits in trade for a shorter menu, but with a large database it may not be intuitive how many names will match a sequence of a few digits.

CASE STUDIES

This section describes three voice response applications that illustrate application of the design considerations and interaction techniques just mentioned. Three different systems all employing speech synthesis for output are discussed. Both the first and third examples are telephone-based information retrieval systems, one for receiving driving directions and the other for reading electronic mail. The remaining system also gives driving directions but operates in real time in an automobile.

Direction Assistance

Direction Assistance [Davis and Trobaugh 1987] is a program that gives driving directions over the telephone.[7] It is an example of the usefulness of speech synthesis in situations where recorded speech would be much more difficult to employ for two reasons. First, Direction Assistance generates fairly complicated textual descriptions of an abstracted data structure, i.e., a route traversing links in a street database. Second, the database of street names is large enough that preparing a recording of each would be tedious. The database does accommodate phonetic spellings for those street names that the synthesizer mispronounces.

To use Direction Assistance, a user calls and enters both source and destination addresses with touch tones. The system then determines a route between these two points and translates the route into a textual description to be read by the speech synthesizer. The motivation for a telephone-based interface is to make this service available while driving around a city and, in particular, when one is

[7]Direction Assistance was written by Jim Davis and Thomas Trobaugh initially at Thinking Machines, Inc., and later at the MIT Media Lab. Davis also used it as a basis for work in intonation at AT&T Bell Laboratories [Davis and Hirschberg 1988].

lost. At such times, access to a computer terminal is unlikely.[8] Spoken directions can be sufficiently effective; a study by Streeter [Streeter *et al.* 1985] found recorded spoken directions to be more easily followed than either written directions or annotated maps.

Direction Assistance includes a street database covering 45 square miles of the Greater Boston area roughly centered on M.I.T. (see Figure 6.5). This map includes portions of a number of the municipalities surrounding Boston. (In Boston several common street names occur in multiple towns on different streets, which is a source of trouble.) This street database has been augmented with information such as street quality, locations of traffic lights and stop signs, and selected major landmarks. The database also notes some other geographic entries, including the locations of traffic circles, railroad tracks, and bridges.

The software components to satisfy these design goals are quite distinct and operate serially. Figure 6.6 shows these program segments. The first part of a user interaction is with the Location Finder, which has the job of eliciting from the caller his or her present location and desired destination. The Route Finder then consults the database to find a path to the destination; this path consists of a series of street segments found in the database. The Describer generates a textual description of the route describing it in terms of the actions a driver must take to follow the path. This text is then read to the user with appropriate pauses by the Narrator.

User Input

The Location Finder handles all user input. It describes the use of the touch tone keys on the telephone and explains that the "*" key is always available for help. It then asks the user for his or her location. All of the prompts are interruptible and type ahead is enabled for all except the yes-or-no questions. The input buffer is cleared whenever the system detects an error, and the error message is *not* interruptible; this is so the error message will not be lost due to type ahead.

In some versions of Direction Assistance, the Location Finder had access to a database linking telephone numbers to street addresses; such databases are commercially available. This access would be particularly useful if the caller were lost and calling from a pay phone, since in most states pay phones are required to display the phone number and address. As street signs are often missing in Boston, it is easy to get into a situation where a driver has no idea which street he or she is on. In telephone number gathering mode, the Location Finder waited until seven digits were entered, looked them up, and pinpointed the location of the caller.

More relevant to our discussion of user input is address mode, in which a house number and street name must be entered. The system uses explicit termination

[8]A company in the San Francisco area installed direction giving machines in public places such as convenience stores, much like video games. These systems used visual menus for selecting a destination and printed detailed driving directions.

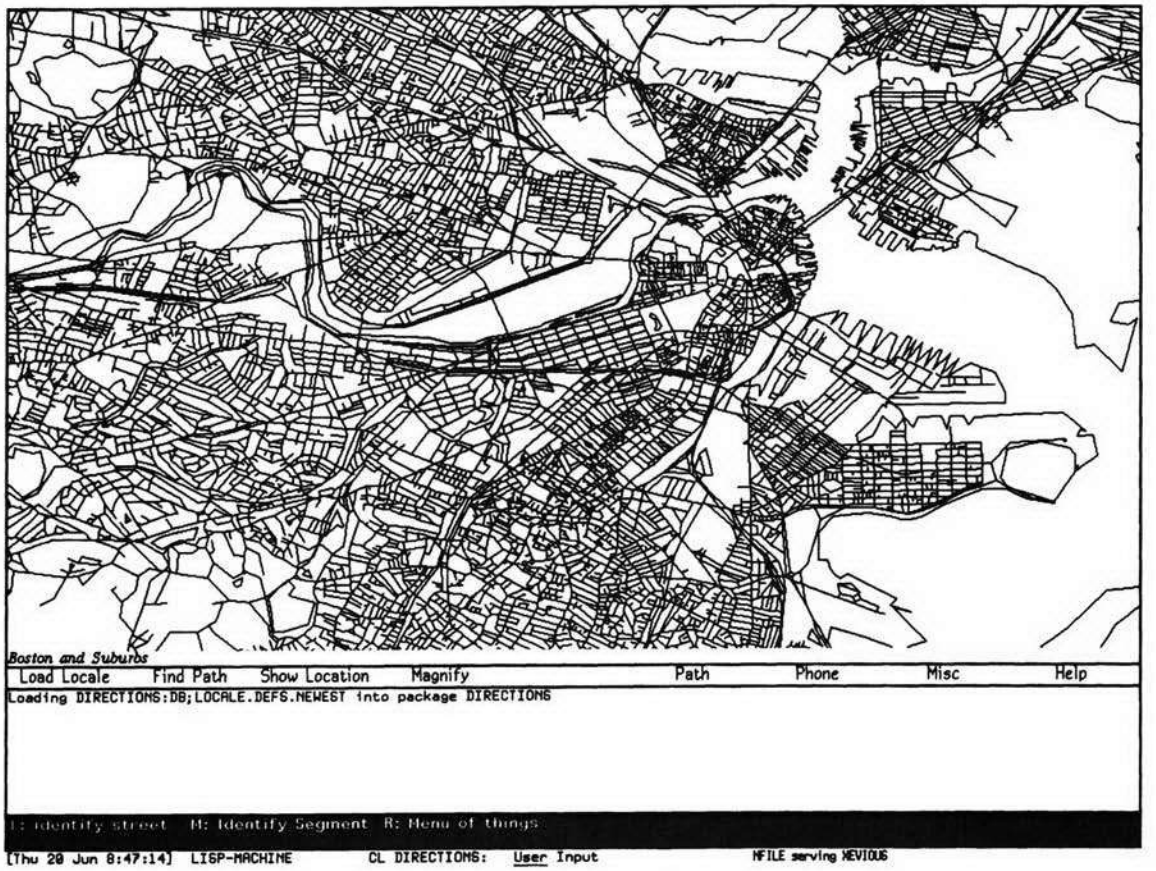

Figure 6.5. The streets in Direction Assistance's database.

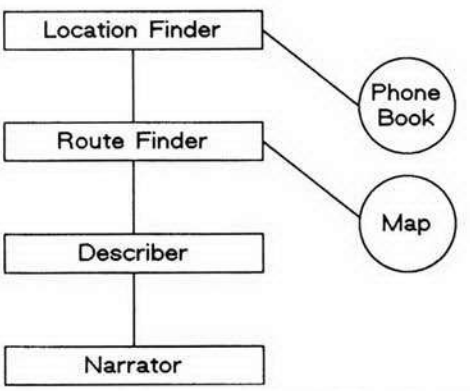

Figure 6.6. Major software modules in Direction Assistance.

for both house number and street name and informs the user to use the "#" key after entering the house number. Of course, explicit termination can fail if the user neglects to use the termination key. If the user pauses, waiting for a response, the program indicates that it thinks the caller is still entering an address and notes that a terminator key is required. If the user forgets the terminator and simply continues, next entering the street name, the system may detect this. When the number of digits entered exceeds the highest known house number in the database, the system indicates that this is an unknown address, reminds the caller to terminate address entry with a pound sign, and starts over.

Entering a street name is more challenging due to multiple instances of the same street name in the database. The caller is prompted to spell the name of the street using one telephone key per character. Again explicit termination is requested. In the case of multiple streets having identical touch tone "spellings," Direction Assistance resorts to a temporal menu of the possible choices. However, much more common than spelling confusion is that the same street may be found in multiple municipalities, or the same name can apply to multiple streets even in one municipality (e.g., Hancock Street, Hancock Place, and Hancock Park in Cambridge). In these circumstances, Direction Assistance again resorts to a temporal menu to finish the selection. To help explain the selection process, the system first indicates why it is confused, e.g., "I don't know whether you mean Hancock Street, Place, or Park. Hit any key when you hear the one you want."

Route Description

Once the origin and destination are known, the Route Finder determines a route. The route consists of a path traversing a number of street segments (each segment being approximately one block long). The route is optimized on the basis of distance, quality of the streets, ease of driving (e.g., a penalty is imposed for left turns), and ease of explanation (a zig-zag route may be shorter but may be more complicated and difficult to follow).

The next step is the most challenging language-generation task of this application. The Describer must take the series of segments from the database and turn it into a coherent description of how to drive to the destination. A list of what to do at each intersection is hardly useful to the driver and interferes with the goal of speaking succinctly. Instead the Describer generates instructions comparable to those a person gives. A passenger assumes the driver takes the "obvious" route (e.g., stays on the main street) until told otherwise. This allows the Describer to compress many sequential segments into a single driving act, e.g., "Stay on Main Street until it merges with Massachusetts Avenue."

The Describer employs a taxonomy of intersection types including such classes as **continue, forced-turn, turn, fork, onto-rotary,** and **exit-rotary.** A software module associated with each is consulted in order at each intersection. The appropriate module chooses to describe the driving act to perform at this intersection. Succinctness is achieved by having the **continue** expert usually choose to say nothing unless a street name changes or a significant landmark is encountered.

The output of the Describer is the text description of a "tour," an example of which follows.

> "If your car is parked on the same side of the street as 20 Ames Street, turn around and start driving. Drive all the way to the end and take a left onto Memorial Drive. After about one quarter of a mile, take the easy left and merge onto Main Street. It becomes the Longfellow Bridge. After you cross the bridge you will come to a rotary. Go about three quarters of the way around it, and turn onto Charles Street. Merge with Storrow Drive."

The text of the tour must be recited by a speech synthesizer; this is done by the Narrator. It is the Narrator's responsibility to recite the tour at a rate that allows the user to write down the directions. Unfortunately people write directions at greatly varying rates; some have elaborate shorthand notations and can record the directions quickly, while others write them word-by-word. The Narrator's only recourse is to pause between each part of the route and wait for touch tone input from the user as a flow-control mechanism, but this does not make for a very comfortable interface. More graceful flow control requires spoken feedback from the direction recipient; an attempt at this is described in Chapter 9.

Back Seat Driver

Back Seat Driver is a subsequent Media Lab project that also employs speech synthesis for direction giving but in the very different context of operating in a car in real time while the user is driving. Although much of the software from Direction Assistance proved useful to this project, Back Seat Driver is different in two respects. First is the motivation to use voice because a driver's eyes are busy watching the road, whereas Direction Assistance focused on remote telephone access. This raises the issue of how to instill user confidence in a system that cannot be seen and speaks only when necessary. Second is the real-time nature of Back Seat Driver; it must know when to speak, how long it will take to say an utterance, and detect when the driver has made a mistake.

Back Seat Driver utilizes a vehicle navigation system to report the automobile's position, speed, and, direction to the route planner.[9] This navigation system uses an inertial guidance system in combination with the constraints provided by the map database (which is stored on CD-ROM) to determine position; for more discussion of the various location determining technologies, see [Davis 1989]. Route planning and discourse generation are done by software running on a computer in the trunk speaking via a speech synthesizer connected to one of the serial ports of the computer (see Figure 6.7).

The initial task of the Back Seat Driver is identical to that of Direction Assistance, namely, to elicit the destination from the driver (the current position of the

[9]The vehicle navigation system was built by the project's sponsor, Nippon Electric Company.

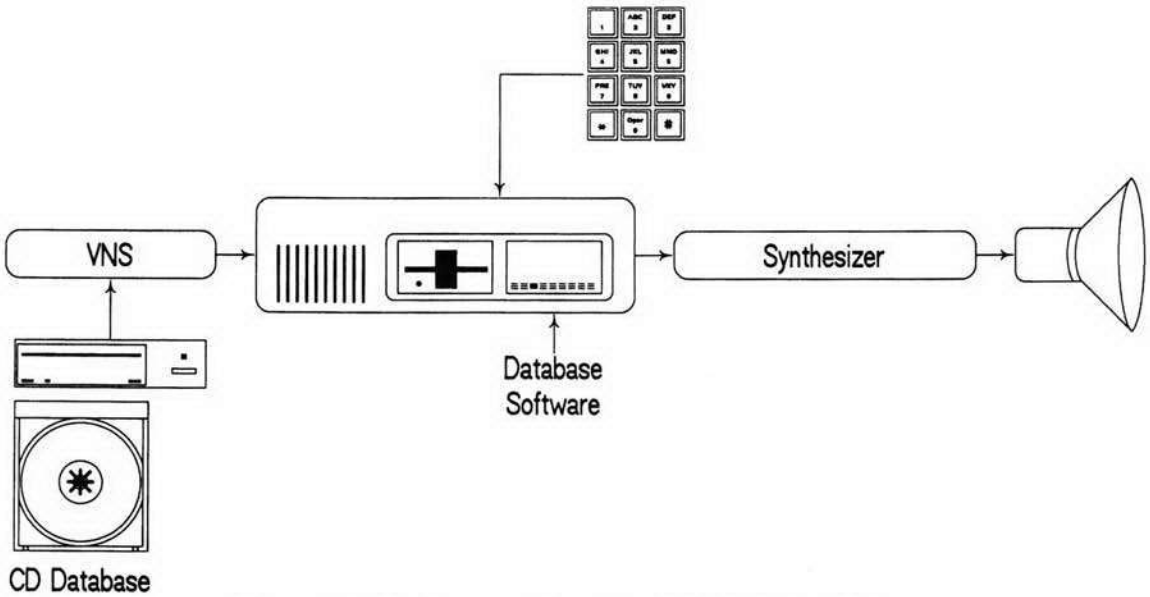

Figure 6.7. Hardware configuration of the Back Seat Driver.

car is known to the system) and plan a route. In various implementations, the destination has been entered by means of a telephone keypad (an interface identical to that used by Direction Assistance) or a computer keyboard. Having obtained the destination, Back Seat Driver plans a route, speaks the initial instruction (e.g., "Turn around and start driving"), and waits for the driver to proceed. From this point on, its task is to provide spoken directions, as they are needed, for each segment of the trip.

The goal of Back Seat Driver is to get the user to the destination; this is facilitated by the secondary task of keeping the driver confident that the system is working and correct. These tasks are closely related, both requiring a richer discourse capability than Direction Assistance as well as an improved method of time management.

Getting the user to a destination requires careful phrasing of the directions. A major problem with the use of synthetic speech for this task is that some street names may not be pronounced correctly. Names are also less useful than other cues in Boston due to the lack of proper street labeling. In an early version, Back Seat Driver would say "Take a left at Fulkerson Street"; this was changed to "Take the next left; it's Fulkerson Street." The driving action is described in a simple sentence for ease of intelligibility [Davis and Schmandt 1989]. The street name becomes a secondary cue, and the drive can still follow the directions even if the name is badly mispronounced or the street sign is missing.

Back Seat Driver also offers advice and warnings to make the drive safer and increase the probability of the driver completing the requested turn. For example, it may say "Get into the left-hand lane because you are going to take a left at the

next set of lights" or warn when a one-way street becomes two way. To provide adequate warning, each step of the directions is repeated twice: once about 30 seconds before the intersection in question and another just before the decision point.

Precise timing of detailed utterances such as "Take a left at these lights" just as the vehicle approaches the intersection, along with the use of deictic pronouns[10] help instill confidence in the driver. Because there is no display and Back Seat Driver does not speak more often than is necessary, it is easy to doubt that it is actually "alive." Utterances of this type affirm that the system is tracking the vehicle's location correctly and will speak when necessary.

To speak at the right time Back Seat Driver needs to manage time in concert with the real world of a vehicle in motion. Time management is achieved by maintaining a series of goals with an estimated time to speak each goal. The time at which an utterance must commence changes relative to the vehicle's speed, and there may not be adequate time to speak all goals. The goals are prioritized so that the most important are spoken first. A goal that would be interrupted by a higher-priority goal (given an anticipated moment to speak the goal) will instead choose not to start as it would be confusing for the system to interrupt itself as well as waste valuable time. Knowing how long it will take to speak a goal enables advance planning and allows some goals to be merged if there is not enough time for each in isolation [Davis and Schmandt 1990]. For example, just before a short jog in the road, Back Seat Driver might say "Take the left at the end of the road, and then you're going to take the very next right."

Goals are dynamic and can change during the course of the trip because the driver does not always follow the directions correctly. Back Seat Driver must cope with these driver errors. It indicates the mistake ("Oops, I meant for you to take a right . . .") but then adapts to the error by computing a new route after telling the driver (". . . but that's OK, I'll figure out a new way to get there"). The Route Finder can operate incrementally while the vehicle is in motion although the driver is told "Slow down while I think about it." All the utterances were carefully chosen during a long iterative design cycle; it is inappropriate to speak in a manner that encourages the driver to simply slam on the brakes possibly causing an accident.

The primary focus of Back Seat Driver has been to experiment with discourse generation in real time so as to determine the appropriateness of speech output for a computer driving assistant. The project was based on the belief that voice could be employed effectively in a real-time application by implementing a goal-oriented discourse model. Although it was never rigorously tested against a display-based system, many users who test-drove the car found it effective and even reassuring. This was achieved in large part by the careful wording and timing of each utterance to match the car's position and speed.

[10]Deictics are pronouns that refer to objects identified by their position relative to the speaker. "This" and "these" refer to nearby objects, "that" and "those" to distant objects.

Voiced Mail

Most telephone-based systems employing speech output involve a process of explicit user selection followed by a system response often repeated step-by-step while the user performs several operations. These systems spend a substantial portion of their time reciting menus. The menus may be hierarchical requiring the user to traverse several submenus and enter multiple key presses to achieve the desired action. While it is likely that a hierarchical structure may be well suited for the novice or occasional user, the experienced user learns the sequences and, provided the menus are interruptible, never hears much of them; callers are still often annoyed to key-press their way through the thick layering of commands to get to the information being sought.

This section presents as a case study a system that took a rather different view of user interaction. Voiced Mail [Schmandt 1984] was a program developed at M.I.T. to allow telephone access to electronic mail using speech synthesis. Voiced Mail provided a streamlined user interface oriented towards experienced users. It tried to minimize the time and attention required to navigate the user interface by avoiding automatic recitation of menus. Instead, the application followed a default sequence of presenting information about each message, then the message itself. The small number of commands were always active, and the application was always interruptible.

Voiced Mail applied extensive filtering to messages to remove extraneous information, made presentation decisions based on the length of each message, and rearranged message order to simplify presentation. These tasks were performed to minimize the time and attention required to use the application. Voiced Mail also employed a repetition strategy to help cope with the vagaries of synthesizing free-form, human-authored text.

Voiced Mail illustrates some advantages of incorporating multiple speech technologies into a single application. In Voiced Mail's later stages, recorded voice was used for generating replies to text messages as the keypad was inadequate to select between a limited repertoire of precomposed text messages. The synergy from employing multiple channels for computer interaction and supporting multimedia databases is a theme to which we will return in Chapter 12.

Flow of Control

To access Voiced Mail, a user dialed in and logged onto the system using his or her computer login ID and a password. The application would scan the user's new mail messages and decide how to present them. A default path was then taken through the messages; with no additional user input, the system would read each message in order to the caller. At no point was the user confronted with an explicit "If you want me to do . . . (some action), press a key"; rather, an active presentation was started, which could be interrupted at any time. There were places where the system would pause a moment to encourage possible user input, but if none was detected it would continue reading messages. This was all part of the strategy of avoiding menus.

Instead of using an explicit menu, Voiced Mail used an implicit set of commands that were always active and interruptible. These commands allowed forward or backward movement among messages or senders and invoked a few other functions. Since there were no explicit menus, each digit on the keypad was assigned to one command (the keypad mapping is shown in Figure 6.8.) This was a principled and deliberate choice because ease of use was more important than the number of functions supported by the application.

Voiced Mail sorted messages according to the sender and then presented all messages from the same sender sequentially, beginning with the sender who had sent the most messages.[11] Most mail systems present messages in the order in which they are received, but time-sequential messages bear little relationship to each other. Sequential messages from the same sender are often related; this ordering tried to preserve such context. A more thorough strategy would have been to identify a "thread" of messages with the same subject line (generated automatically by the "reply" command found with most text-based electronic mail systems) but from various senders.

After Voiced Mail determined presentation order, messages would be played one by one. If the message was brief, it was played immediately. If it was long (as measured by number of words), the system would warn "It's a long message, about . . ." and recite the subject line. The system would then pause to invite user input (for example, to skip the long message). In the absence of any input, the message body would be played.

[11]It would have been useful to have included an optional user profile that identified important message senders by name and played their messages first. This is an aspect of the filtering approach mentioned later and was implemented in a later email reading program.

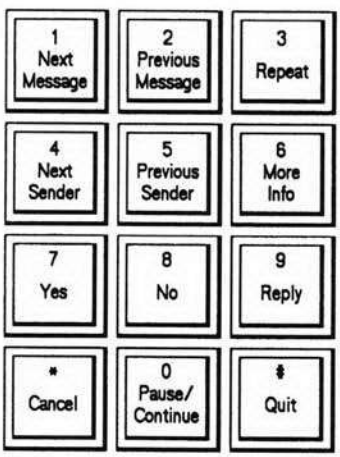

Figure 6.8. The command set for Voiced Mail.

This design was based on the observation that it would take more time to ask the user "Do you want to hear this message?" and wait for a response than to just play the message. The subject was not recited for the very short messages, as it was often redundant to the message body. This, however, was a mistake for some messages, e.g., that shown in Figure 6.9. A sample user interaction is shown in Figure 6.10.

Text Presentation Strategies

Because it avoided menus Voiced Mail spent most of its time actually reading messages. Recitation of a message required some careful filtering and an effective presentation strategy. There is much useless header information in a mail message. There may be punctuation or spacing conventions that should be preserved if possible, e.g., a double space between paragraphs is "spoken" as a short pause. Since there may also be typographic errors or words in the message that the synthesizer mispronounces or the user may need to hear a sentence again, a single linear pass through the message may be insufficient: the presentation must be interactive.

Figure 6.11 shows the full text of an electronic mail message including the rather voluminous header. Many fields, such as the message ID and the chain of hosts through which the message passed during transmission, are pointless to synthesize so they must be removed. Voiced Mail went further and decided that the only fields in the header that were vital were the sender and possibly the subject. A "more info" key offered more information, such as the full name of the sender and when the message was received. The time data was processed to produce text such as "Yesterday morning at 10" instead of "Wednesday 25 August 1993 9:57:32 EDT," which both takes longer to speak and is more difficult to follow.

For the sake of its repetition strategy (see below), Voiced Mail parsed the message into sentences. To aid clarity, a pause was introduced between each sentence; this pause was about twice as long as would ordinarily be produced by the synthesizer. A pause twice as long again was used at each paragraph in the message body as the paragraph structure is also important for understanding the message's content.

```
Message 141:
From derek Thu Mar 15 18:05:25 1990
To: geek
Subject: Metal-Oxide Varistor
Date: Thu, 15 Mar 90 18:05:21 EST

It's here . . .  I'll leave it in the sound room on the middle shelf . . .
```

Figure 6.9. A mail message for which the subject is required to understand the message.

```
"Welcome to Voiced Mail. Please log in."

user presses 4-3-3-5 (sequence for "geek", the user's login).

"Hello Chris Schmandt. You have twelve new messages. <pause>
Four from Walter Bender. <pause>
Message one. This is a very long message; it's about window system
subroutines . . ."

user presses 1 (next message).

"Message two. It's about the new conference room. <pause>
Can we get together tomorrow afternoon to decide about chairs?"

user presses 7 (affirmative reply).

"Message sent. Message three. It's about Monday's demo . . ."

user presses 4 (next sender).

"Two messages from Ben Stoltz. <pause>
Message one. It's about the driver bug . . ."
```

Figure 6.10. A sample Voiced Mail session.

Voiced Mail compromised intelligibility for speed by playing text somewhat faster than the default synthesis rate specified by the synthesizer hardware. At any moment, the user could invoke the **repeat** command. This command would cause the sentence currently being spoken (or which had just been spoken) to be repeated at 75% speed. If **repeat** was invoked again on the sentence, the sentence was spelled letter by letter.

Spell mode is so slow as to be nearly useless, however, and it is also quite difficult to follow. An alternate strategy, developed shortly after Voiced Mail, passes the message body through the local spelling checker dictionary. Those words not found in the dictionary are spelled. This will catch most typos as well as many of the proper nouns that might be particularly challenging to synthesize. For example, the phrase, "Schmandt debugged the sotfware yesterday morning" would be pronounced "S-c-h-m-a-n-d-t debugged the s-o-t-f-w-a-r-e yesterday morning."

Advantages of Multiple Media

One of the advantages of electronic mail is the ease of posting a reply to the sender of a message. While reading mail, one is usually positioned at a screen and keyboard, which also provide the user interface for typing a reply. A single

```
From dcj@Sun.COM Tue Sept 22 15:53:21 1991
Received: by media-lab (5.57/4.8)  id AA11928; Tue, 22 Sept 91
15:53:15 EDT
Received: from snail.Sun.COM (snail.Corp.Sun.COM) by Sun.COM (4.1/SMI-
4.1)
id AA24516; Tue, 22 Sept 91 12:53:51 PDT
Received: from jacksun.sun.com by snail.Sun.COM (4.1/SMI-4.1)
id AB18658; Tue, 22 Sept 91 12:51:56 PDT
Message-Id: <9109221953.AA06751@jacksun.sun.com>
To: Chris Schmandt <geek@media-lab.media.mit.edu>
Subject: possible bug in fax.c
In-Reply-To: Your message of Tue, 15 Sept 91 12:43:41 -0400.
            <9109151643.AA27528@media-lab>
Date: Tue, 22 Sept 91 12:53:13 PDT
From: dcj@Sun.COM
Status: RO

I received your sound attachment email just fine.
This is a nice addition to Phoneshell!
```

Figure 6.11. An electronic mail message including its full header.

keystroke to the mail reader ("R" for "Reply") initiates a reply with the return address generated automatically. Since the Email medium is asynchronous, the sender need not worry whether the recipient is awake to receive the message in whatever time zone the destination lies.

It was important for Voiced Mail to support a reply mechanism as well, so three of the twelve telephone keys were devoted to reply commands. One key would generate a message of the form "I read your message about . . . My answer is yes" and another generated a similar negative reply. A third key prompted the user for a phone number and sent a "please call me at . . ." message. Unfortunately, replies are rarely so unconditional as a simple yes or no. Also, when traveling one's location and hence phone number are often transient, so the reply mechanism was enhanced to include the ability to record a voice message. The party who sent the original message would receive a computer-generated text mail message containing instructions on how to access the reply, which would involve calling the Voiced Mail telephone number and entering a special code at the login prompt. Upon hearing the message, the caller could then leave another voice recording in reply.

Of course, the recipient of the recorded message might be another subscriber to Voiced Mail. In this case, while reading mail from a terminal the recipient would see the same text message as delivered to the non-subscriber, i.e., he or she would have to call in to hear the voice message. If the subscriber were reading mail with Voiced Mail, the text message would never be seen as the system would simply play the recording. Such a unified interface to both voice and text messages was extended and incorporated into the Phone Slave conversational answering machine described in Chapter 11.

An important advantage of merging voice and text messages is the elimination of the need to consult multiple devices (telephone for voice mail, computer terminal for text mail) to find out whether any new messages have arrived. In the case of Voiced Mail, the initial motivation for including voice messages was the new reply capability required by telephonic access; text messages could be turned to voice, but voice replies could not be converted to text. This, therefore, is a case in which a new user interface medium results in the need to support multiple media in the underlying database being accessed.

Voiced Mail Today

Voiced Mail addressed universal access to electronic mail messages. At the time it was built (1983) modems were still 300 baud and computer terminals at home uncommon even at a research institution like M.I.T. But even under these circumstances, synthesis was too tedious to use for longer messages. Instead, users would scan messages to see if they had anything important to deal with (such as scheduling lunch) before coming into work. The exception was when users were traveling and had no other access to their mail.

Modem speeds have increased and terminals or personal computers at home are now commonplace; this decreased the need to use speech synthesis to access mail messages at home (although we do like to joke about hearing mail while in the shower). An interface such as Voiced Mail is more valuable to mobile users who may want to call in from airport lobbies or roadside phones. Voice is also particularly useful when used while doing other activities. One example is using a cellular telephone to scan mail while driving to work (the morning commute otherwise being wasted time.) A safer version of this scenario would involve uploading email as text into a laptop computer over telephone lines in the morning and then synthesizing it in the car while using speech recognition (instead of touch tones) to navigate among messages.

Speech synthesis access to Email over the telephone remains in surprisingly high demand at the Media Lab, and it has been included in a more integrated messaging environment. Phoneshell, which as described in Chapter 12, brings increased functionality to Email access including the ability to record a voice reply, to type a text reply using the telephone keypad, to send copies of replies to multiple parties chosen from an address listing, and to send a message to a third party as a fax. Another feature of the newer version is the ability to *filter* mes-

sages into broad categories (e.g., "urgent," "important," "personal," or "junk") based on pattern matching against email header fields (such as sender and subject) and the message body. Email filtering was first introduced by Malone in the Information Lens and has been shown to be an effective means of managing mail [Malone *et al.* 1987, Mackay *et al.* 1989]. For Phoneshell users, filtering makes it possible to attend to the most important messages first, which is essential when dealing with a slow communication channel.

Although the ability to reply and sort messages is crucial for the current success of speech access to email, the primary reason Email is popular is simply the ease of dialing in. Although most of Phoneshell's present users possess computers and modems at home, telephone access allows casual and low-overhead message retrieval from locations such as stores, pay phones in airports, other offices on campus, or even the kitchen when one merely wishes to check on message status. Even if email is automatically sent to alphanumeric pagers (this is done at the Media Lab and many other locations), each page is limited to several hundred characters; this asynchronous notification provides even greater motivation to immediately call in from the nearest telephone to hear the entire body of an urgent message and possibly send a timely reply.

SUMMARY

This chapter has discussed the advantages and disadvantages of voice from the perspective of interactive systems employing voice response. Speech output may be difficult to employ effectively for information providing because it is slow, temporal, serial, "bulky," and may not be private enough for some office situations. At the same time speech is advantageous over other media because it can be used at a distance over the telephone or heard across a room or when a user's hands and eyes are otherwise occupied.

A number of design considerations were suggested based on comparing the advantages to the disadvantages of speech output and how it should be employed in an application. Some applications may simply be inappropriate for speech output. For appropriate applications, speech may be sped up to increase performance, but a means of repeating an utterance may be needed to clarify messages when a user fails to understand them. There may be a role for multiple voice styles in a single speech-output application.

Because many interactive voice response systems are telephone based, we paid particular attention to the issue of user input with the telephone keypad. Touch tones can be used to select an item from a list, to answer yes-or-no questions, or to make a selection from a menu. Implicit or explicit termination of input may be more suitable to different tasks being performed over the telephone.

Three case studies were presented to illustrate the points being made in this chapter. Direction Assistance employs a variety of input techniques to trigger discourse generation from a complicated computer database of street segments. Back Seat Driver, using the same database, must generate directions in real time

and instill confidence in the user that it remains on target despite periods of silence—a need not present in graphical interfaces. Finally, Voiced Mail illustrated how text filtering and organizational issues can minimize the amount of information that must be attended to while reading electronic mail with speech synthesis; it also eschewed prompts and menus in favor of a default flow of control during a session with a caller.