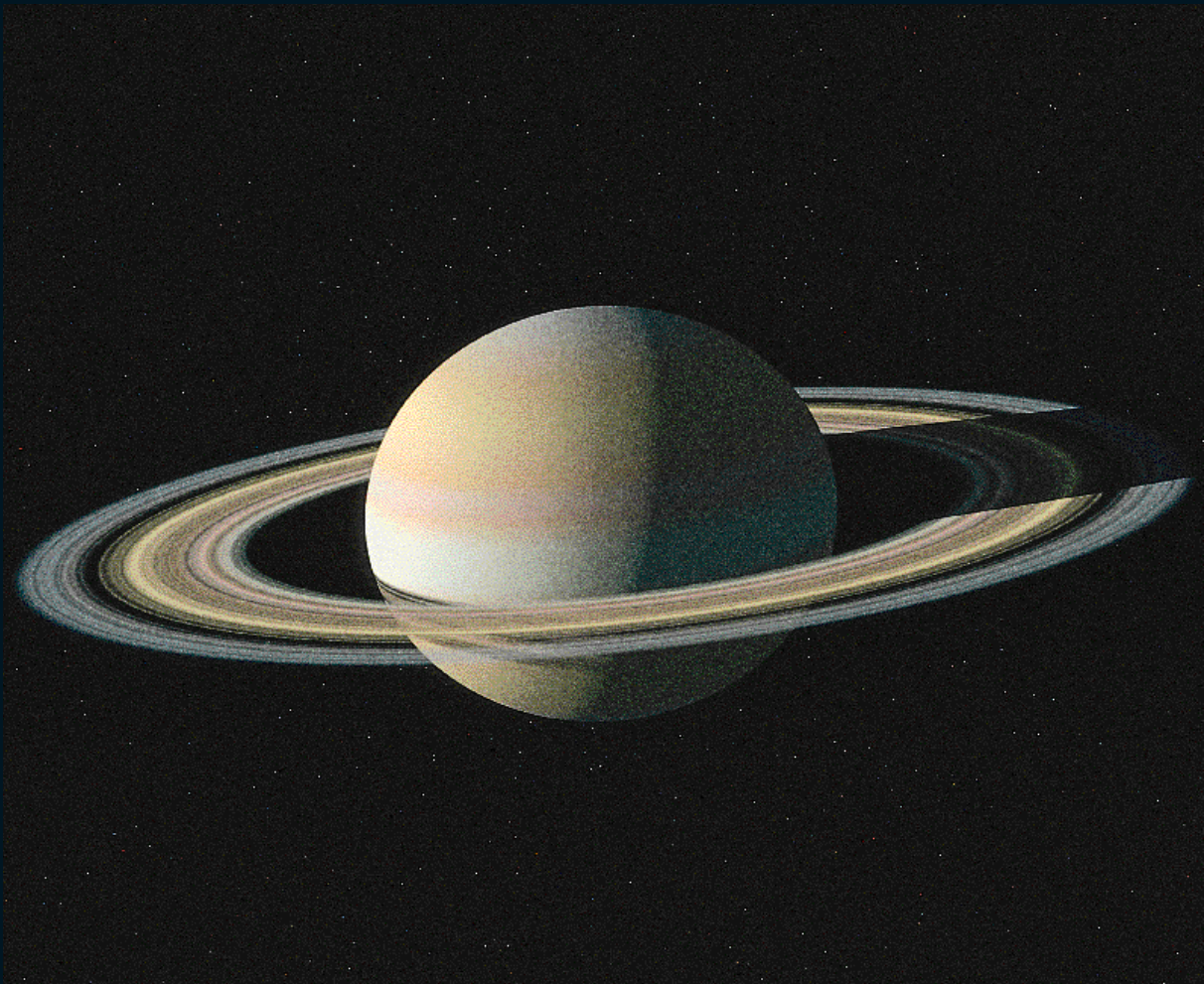


NICOLE YANKELOVICH

How Do Users



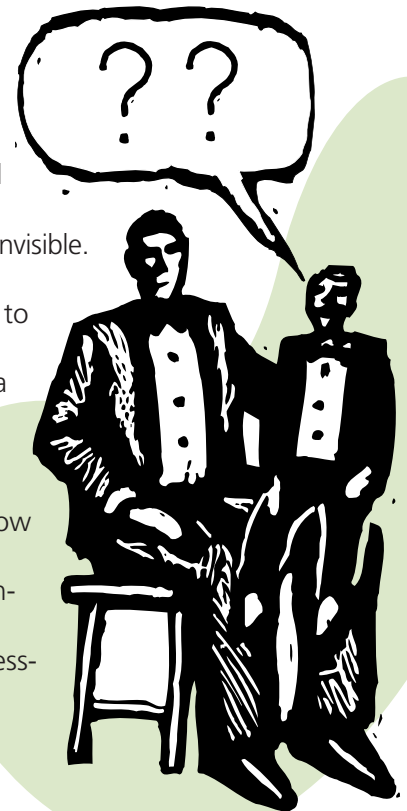
SATURN

Know What To *Say?*

As speech recognition technology improves, many software designers are being challenged to design speech user interfaces. For me, one of the most challenging issues is “How do users know what they can say?”

If you think about it, a speech-only interface poses the same problems as a command-line interface. The functionality of applications is hidden, and the boundaries of what can and cannot be done are invisible.

Graphical interfaces were invented largely to make hidden functionality visible to the user. In a speech-only environment, such as over a telephone, it is not possible to display menus, show options, or highlight buttons. Instead, other techniques must be used to guide users through a successful interaction.



To gather insight into ways of tackling this difficult problem, I asked a dozen respected speech user interface designers (see Figure 1) to give me examples of how they let their users know what they can say.

Defining the Problem

Before discussing the solutions, let me provide you with some additional insight into the problem. One of the applications that I designed is a speech-only interface to a shared calendar [4]. Here's what an interaction over the telephone with the SpeechActs Calendar application might sound like:

Computer: Currently, you have "lunch with Julie." Later, at 2:30, you have "Staff Meeting in Conference Room B."

User: What's on my calendar the Monday after Easter?

Comp: On Monday, April 8, you have "Call Fidelity Monitor," you have "Barbara Munch's birthday."

User: What does Paul have?

Comp: On Monday, April 8, Paul has "Chris goes to Wisconsin." From 10 o'clock to 10:30, "Meeting with Bob."

User: What about the next day?

Comp: On Tuesday, April 9, Paul has no appointments.

The dialogue has progressed smoothly so far, but now the user gets into trouble:

User: What do I have tomorrow morning?

Comp: Sorry?

Although this query about "tomorrow morning" sounds perfectly reasonable given the prior dialogue, the user has just crossed the functional boundary of what the application supports. In designing the calendar application, we opted to support browsing calendars only in units of whole days and handle "out of grammar" requests, such as this one, with our error recovery interface. The decision to constrain the grammar was driven primarily by our desire to maintain reasonable recognition performance. Perhaps the decision seems arbitrary, but it's a slippery slope. Once you allow users to talk about "morning," "afternoon," and "evening," they'll expect to be able to say things like "early afternoon," "after 6 p.m.," "around noon," and so forth. At some point, the constraints of the speech recognition technology dictate that only a limited set of utterances be allowed. Designers must draw the line someplace.

The calendar example illustrates one side of the problem: Users assume they can say things that the application does not support. The flip side of the problem is that users don't know to say things that the application does support. Here's an example from another one of my group's projects: an Automated Customer Service Representative. With this application, users can view the Lands' End clothing catalogue on an interactive television display. When they dial a telephone number, they converse with the Automated Rep, which helps them browse the catalogue and select items for purchase. When the top level display is presented (Figure 2), the user hears

Comp: Thank you for calling Lands' End. Which of these items are you interested in?

The user will do fine by answering the question directly:

User: I'm interested in women's clothing.

The application, however, will accept a much richer set of responses. These range

FIGURE 1 *Contributing speech user interface designers*

Adam Cheyer	SRI Artificial Intelligence Center
Stephan Gamm	Philips Research Laboratories
Francis Ganong	Kurzweil Applied Intelligence
Jim Glass	MIT Spoken Language Systems Group
Caroline Henton	Digital Equipment Corp (formerly of Voice Processing Corp)
Candy Kamm	AT&T (formerly of Bellcore)
Troy Kamphuis	Nuance Communications
Demetrios Karis	GTE Laboratories
Tony Lovell	Wildfire
Amir Mané	AT&T
Matt Marx	Applied Language Technologies (AlTech)
Patti Price	SRI



FIGURE 2
 First screen of interactive television display for Lands' End catalogue. (Courtesy of Lands' End Direct Merchants)

from specific...:

User: I'd like a large men's cotton button-down shirt.

...to quite vague:

User: Do you have any red sweaters?

One solution, of course, is to create a much more detailed initial prompt that spells out the range of options available to the user. The disadvantage of this approach is that speech output is slow and temporal. Not only are long prompts costly in terms of the time they take, but users often only remember the end of what was said. In the following discussion, hints, suggestions, and other techniques are described that attempt to provide more satisfying solutions to the problem of letting users know the range of available options.

To summarize, the problem of knowing what to say to a speech application has two components. Users can assume the computer will be able to understand more than is actually possible, and users can be unaware of functionality that is available.

Design Constraints

Like all design problems, the techniques available to guide users in speaking well-formed input are subject to technological constraints. A major constraint on designing a speech application involves the features supported by the speech recognizer being used for the project. Some recognizers can support a fairly large vocabulary of continuous speech, whereas others can support only a small vocabulary (but typically with greater reliability). Other recognizers require discrete words or phrases rather than continuous speech. In addition, some recognizers support a feature, important to interface design, called "barge-in." This refers to a user's ability to interrupt speech output and still be understood by the speech recognizer.

A second design constraint is the environment in which the speech recognizer is being used. Many speech applications are designed to work over a telephone, but others are designed to be used in conjunction with a graphical interface or as part of a consumer device.

Another constraint involves the user profile. Some speech applications are intended for

Following are definitions of speech-related terms used in this column as well as some suggestions for learning more about speech technologies.

Some Speech Technology Terms

Speech Recognition

The capability of a computer to take spoken input from the user and convert it to text.

Speech Recognizer

A software system (sometimes with hardware components) that processes a digital audio signal from a file, microphone, or telephone. The processing produces the best fit of possible word sequences based on a frequency table or predefined grammar specification. The output is one or more strings of text, often with associated measures of how closely the string matches the specification.

Discrete Speech

Some speech recognizers only allow users to speak one word or short phrase at a time, such as "Calendar" or "Send-to-back."

Continuous Speech

Other speech recognizers allow users to speak connected streams of words, although usually only one sentence at a time. For example, "I would like my calendar," or "Send this to the back."

Vocabulary

The words or phrases a speech recognizer can "hear." Both discrete and continuous speech recognizers have a word list, sometimes called a "lexicon."

Grammar

A specification used by some continuous speech recognizers for how words in the vocabulary can be strung together.

Dictation

The use of speech for entering free text, as in a memo or the body of an electronic mail message. Commercially available dictation

systems currently all use discrete speech recognition, but with very large vocabularies. Dictation systems are distinct from "command-and-control" systems, which are designed to allow the user to issue commands and to control application behavior.

Speech Output

The computer can either play recorded audio messages or can convert text to speech using a speech synthesizer. The recorded audio provides higher quality output, but a synthesizer is almost always used when the content of the output is not known ahead of time.

Prompt

A recorded or synthesized message produced by the system for the user.

Speech-only Interface

An application interface that has no other input or output mechanism other than speech. Telephone-based applications are the most common speech-only interfaces.

Multimodal Application

In the context of this article, an application that uses any number of input and output modalities, including speech.

Books that include some speech design issues:

Interactive Speech Technology: Human Factors Issues in the Application of Speech Input/Output to Computers. Baber, Christopher, and Noyes, Janet M. (eds.), Taylor & Francis Ltd., London, 1993.

Schmandt, Christopher. *Voice Communications with Computers.* Van Nostrand Reinhold, New York, 1994.

infrequent use by a large user population. For example, a number of phone companies offer speech recognition for accepting or rejecting a collect call. Other speech applications are designed for frequent use by a small user population, and others fall somewhere in the middle of this spectrum.

Techniques

The techniques I focus on for helping users of speech applications produce well-formed spoken input all involve design of prompts. Although other techniques such as printed documentation and on-line help can be indispensable, prompt design is at the heart of effective speech interface design.

Matt Marx of AlTech offers some useful vocabulary for analyzing different strategies for prompt design. He suggests that prompts fall along a continuum from implicit to explicit. Troy Kamphuis of Nuance Communications illustrates three points along this continuum with these possible prompts for a banking application:

Comp1: Welcome to ABC Bank. What would you like to do?

Comp2: Welcome to ABC Bank. You can check an account balance, transfer funds, or pay a bill. What would you like to do?

Comp3: Welcome to ABC Bank. You can check an account balance, transfer funds or pay a bill. Say one of the following choices: check balance, transfer funds, or pay bills.

Implicit prompts, such as the first prompt in the set above, are usually inappropriate for small vocabulary recognizers because they invite open-ended responses.

Technique: Explicit Prompts

Most designers constrained by small vocabularies stick to the explicit end of the spectrum, illustrated by the third prompt in the set. The most explicit prompts are called directive prompts [3]. These tell the user the exact words they should say. Amir Mane of AT&T points out that listing options in this way is most effective if the recognizer supports barge-in. He provides this example:

Comp: Please choose: leave a message, return my call, or talk to the secretary.

While the prompt is playing, the user can interrupt and say one of the choices. Since users can say the option as soon as they hear it, this technique mitigates the problem of users forgetting the exact wording of the choices by the time a long prompt ends. It also means, however, that the user may not hear all the possible choices. This problem is not so bad if the list is short, as in the preceding example, or if it acts as a reminder, as it does in AlTech's prototype Music Mall application. Notice in this example that all the prompts are explicit, but the first is less explicit than the second two:

Comp: Please say the name of the artist you want.

User: The Beatles.

Comp: I have a lot of albums by the Beatles; please say the name of the album you want. For example, you could say "Magical Mystery Tour" or "Abbey Road." For a full list of albums I have by The Beatles, say "list."

User: List.

Comp: Here are the albums I have by the Beatles. Say the one you want as soon as you hear it. "With the Beatles." "Beatles For Sale." "Revolver." "The White Album." "Please Please Me...."

User (interrupting): The White Album.

Comp: I think you said, "The White Album." Is that correct?

User: Yes.

Comp: Okay. To sample "The White Album"....

The AlTech recognizer used in the Music Mall application is capable of handling a large vocabulary (more than 2,300 artist names). From a technical standpoint, the recognizer could also handle a flexible grammar, but the designer made the choice to tightly constrain the user input instead of allowing more flexible input such as:

Comp: Hey man, I'd like to hear the Beatles, okay.



Nicole Yankelovich
Sun Microsystems
Laboratories
+1-508-442-0441
nicole.yankelovich@
east.sun.com

DESIGN COLUMN EDITOR

Kate Ehrlich

Research Scientist

Lotus Development

Corporation

One Roger Street

Cambridge, MA 02142

+1-617-693-1899

fax: +1-617-693-5541

Kate_Ehrlich@crd.lotus.com

The constrained approach significantly helps to minimize errors, which is essential for an application that has to work well for first-time callers.

Technique: Implicit Prompts

Although flexible grammars, coupled with large lists of options, tend to increase error rates, this type of input has an enormous potential benefit to users. Most people working on research-oriented applications are attempting to design some degree of conversational interaction. Perhaps the best examples of conversational speech interfaces are those developed as part of a competition funded by ARPA to create a speech interface to an air travel information system. In many of these systems, the dialogue with the user is a mixed-initiative interaction. Here's a hypothetical example from the MIT Galaxy air travel system:

User: I'd like to fly from Boston to London.

Comp: What date will you be traveling on?

User: I want to leave next Friday.

Comp: Here are the flights from Boston to London on Friday, May 31 ...

Notice that the user begins the conversation. In this example, the user does not provide a fully qualified request. If some information is missing, the application formulates a query to collect the missing data. The application would have accepted a request such as:

User: I'd like to fly from Boston to London next Friday afternoon on British Airways.

or

User: Tell me about British Airways flights leaving Boston for London a week from Friday.

or

User: Is there a flight next Friday on British Airways from Boston to Heathrow in London?

The grammar for this application allows a wide range of grammatical constructs. In addition, think about the list of options in every place in the sentence that a city name, airline, or date can go. This application probably has too high a perplexity (number of equally likely word choices at a given time) to be practical over today's fairly low-quality telephone lines. Because of this current technological constraint, most people designing conversational interfaces use cleverly designed prompts to encourage users to speak sentences that conform to a somewhat more constrained grammar. The following example from MIT's

Wheels classified advertisements application illustrates the use of a suggestion:

Comp: There are 100 convertibles. Can you specify a make or a make and model?

Users are not required to specify a make or model at this prompt. The system will accept queries such as

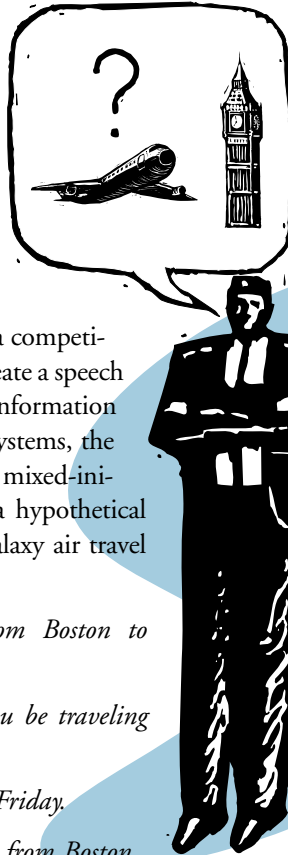
User: I'd like a red one with a manual transmission.

If they don't know what to say, however, the suggestion helps the user to formulate a query that the system is likely to understand. Another common technique on the implicit end of the spectrum is to use subtle discourse cues in the prompts. In conversations, people tend to mimic the style of their conversational partners in terms of vocabulary used, length of utterance, and grammatical constructs. For example, in the SpeechActs Calendar, the user might ask about a person whose name is ambiguous:

User: What's on Bob's calendar tomorrow?

Comp: Did you mean Bob Kuhns or Bob Sproull?

User: I meant Bob Sproull.



Web sites with general information about speech technology:

Comp. Speech FAQ <http://www.speech.su.oz.au/comp.speech/>
Commercial Speech <http://www.tiac.net/users/rwilcox/speech.html>
Speech Toys <http://www.speechtoys.com/spchtoys/>

Speech demonstrations over the telephone:

Note: When calling these numbers, listen to the prompts and think about whether and how they could be improved.

CheckFree Corporation
(800) 392-0743
Electronic bill payment.

Linkon Demonstration Hotline
(800) 793-3667
Voice fax on demand and text-to-speech demos using Lernout & Hauspie recognizer.

Nortel StockTalk
(514) 765-7862
Speech system for stock quotations.

Voice Control Systems (VCS)
(214) 404-9405
Alphabet Recognition Demo

VCS Barge-In
(214) 404-0777
Demonstrates user's ability to interrupt speech output.

VCS Connected
(214) 490-0767

Connected Digit Recognition Demo

VCS Credit Card
(214) 490-1210
Credit Card Validation Demo

Voice Processing Corporation (VPC)
(617) 577-8422
Demos include Voice Dial, Auto Attendant, Credit Card, Continuous Digits

Wildfire Communications, Inc.
(800) 945-3347
Not much speech recognition in the demo, but you can listen to a simulated session between a Wildfire user and the system.

Welcome to ABC Bank. You can check an account balance, transfer funds, or pay a bill. What would you like to do?



This “Did you mean...” disambiguation prompt encourages users to answer with just a name or with something like “I meant...” or “I mean...” It is certainly possible for the user to say something outside the grammar, but the aforementioned prompt makes it much less likely that the user would say something like:

User: Sproull's the one I want.

So far, we have examined purely explicit or purely implicit prompt designs. Not surprisingly, some of the most effective prompt design techniques fall somewhere in the middle of this continuum.

Technique: Incremental and Expanded Prompts

An incremental prompt involves providing an implicit prompt, pausing to wait for user input, and then providing a more explicit prompt if the user does not provide input or says something that cannot be interpreted. Here is how this technique could be applied to the banking prompts described earlier:

Comp: Welcome to ABC Bank. What would you like to do?

User: (silence)

Comp: You can check an account balance, transfer funds or pay a bill. What would you like to do?

User: (silence)

Comp: Say one of the following choices: check balance, transfer funds or pay bills.

For some applications, it is not possible to provide users with an exhaustive list of everything they can legally say. This holds true for GTE's telephone directory system, Speech Connect. In this application, information about several thousand businesses is available. Because of the large vocabulary, the incremental prompt guides users more than it provides them with exact utterances:

Comp: Welcome to Speech Connect. What business please?

User: (silence)

Comp: Please say either the name or the type of

business you are calling, or say "list" for a list of business types or "help" for a help message.

A slight variation on incremental prompting is prompt expansion. In this case, the user responds to the implicit prompt by saying something that the system cannot interpret. The next prompt she hears is an expanded or more explicit version of the first. Here's an example from a telephone answering machine application designed at Philips Research Laboratories [1]:

Comp: You have heard all messages, please give an instruction.

User: Mmmh, I don't know.

Comp: The accepted speech commands are "replay," "delete," "new announcement," and "turn off."

Voice Processing Corporation's Phone-Wizard telephone answering machine application uses a somewhat less explicit prompt to follow a recognition error:

Comp: Who's calling, please?

User: I'm calling long distance. This is Jack Armstrong.

Comp: Please say your first and last names.

Incremental and expanded prompts provide help for users in a way that does not slow down the interaction if the user provides well-formed input at the outset.

Technique: Tapering

Another technique, called tapering, is aimed at shortening the interaction for users as they gain experience with a system. The user is first given an explicit prompt. Later in the interaction, the prompt is tapered to something usually more implicit. In a voice directory application designed at Bellcore, users first encounter this prompt:

Comp: Please say the first and last name of the person you want.

If the user stays on the line to make a second request, the prompt is shortened to

Comp: Say the name of the person you want.

For additional information on SpeechActs, see <http://www.sunlabs.com/research/speech/>.

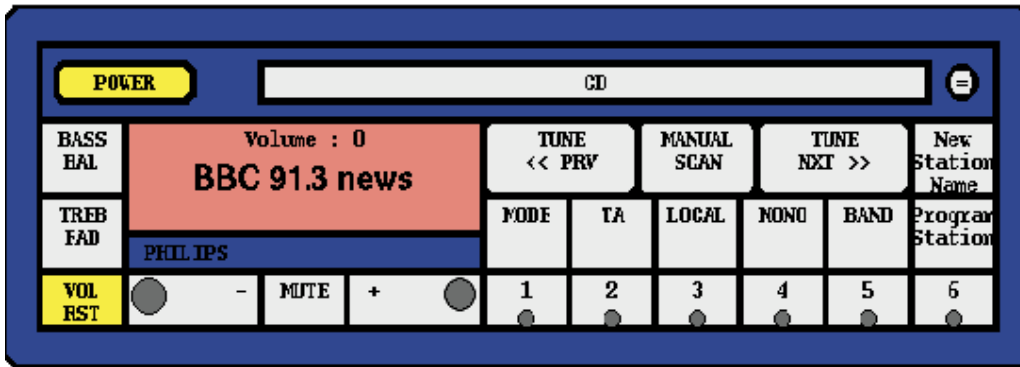


FIGURE 3
 Sketch of Philips Research Laboratories' speech-controlled car radio.

All the SpeechActs applications that my group has designed make extensive use of tapering. For example, in the SpeechActs electronic mail application, users can record outgoing messages:

Comp: Please record your message after the tone. Pause for several seconds when done.

Since many users record multiple messages during a session, this lengthy instructional message would start to sound repetitive very quickly. The next time they ask to record a message in the same session, the prompt is shortened to:

Comp: Record then pause.

Technique: Hints

A variation on tapering is use of hints. For example, novice users of Wildfire's telephone assistant are provided with prompts that involve an implicit, conversational question, followed by an explicit hint.

Comp: Hello. You've reached the office of Tony Lovell. If you are he, press the star key. Otherwise, please say your full name.

*User: (presses *)*

Comp: What's the passcode? Say the digits one by one or use touchtones followed by the pound key.

User: 4 5 4 5 1 2.

Comp: Oh, hi! You have 2 new messages. The first is from Rich Miner. I'd recommend saying What's-it-say, Describe-it, or Next-item.

User: What's it say?

Comp: (message is played) I'd suggest Give-

them-a-call, Send-a-reply, or Throw-it-away.

User: Give them a call.

Comp: Rich Miner at work? yes or no?

User: Yes.

Comp: Dialing... press 44 when you want to end the call.

Hints are different from explicit prompts in two ways. Rather than comprehensively listing all choices, they include only a few of the most useful or common options. Also, hints are designed to be removed. The Wildfire system keeps track, across sessions, of how many times a user has successfully responded to each individual prompt in the system. After a certain number of successful attempts, the hints are removed from that prompt, leaving only the implicit question that the user has already become accustomed to hearing. The previous dialogue would be much shorter for a frequent user of the system:

Comp: Hello. You've reached the office of Tony Lovell. Please say your full name.

User: It's me Wildfire.

Comp: What's the passcode?

User: 4 5 4 5 1 2.

Comp: Oh, hi! You have 2 new messages. The first is from Rich Miner.

User: What's it say?

Comp: (message is played)

User: Give them a call.

Comp: Rich Miner at work?

User: Yes.

Comp: Dialing...

This personalized approach provides a powerful means of offering help for commands that are infrequently used and tapering prompts that are frequently encountered.

Techniques Specific to Multimodal Systems

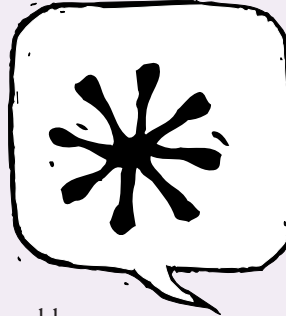
Thus far, the discussion has focused on speech-only systems. Although letting users know what they can say may not be as problematic in a multimodal application, the problem nonetheless exists.

In both consumer devices and desktop dictation systems, one strategy is what Francis Ganong from Kurzweil Applied Intelligence calls “say what you see.” In the Kurzweil Voice for Windows dictation product, the rule of thumb is that users can say any command that is visible in a menu or dialog box. In a voice-controlled car radio designed at Philips Research Laboratories [2], for example, users can speak the names of the labels on the dedicated buttons for “bass,” “treble,” or “volume.” Another technique is to use the display on the radio to give users clues about acceptable command words. For example, the display always shows the name of a station’s category, such as “news” (Figure 3). This may help suggest to users that they may speak names of other categories such as “pop” or “classical.”

A larger display, such as the one available in a dictation system, allows the designer to provide users with an explicit list of all the currently available command words and phrases. The Kurzweil Voice product shows a scrollable list of currently active words, organized hierarchically. This technique is most effective when the word list is relatively short.

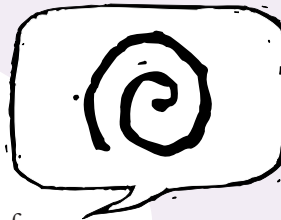
Adam Cheyer of SRI’s Artificial Intelligence Center describes a somewhat different approach for educating users about

what they can say in a multimodal application. His approach is to “encourage a transition towards spoken or multimodal interactions through complementary audio and visual feedback. When users execute actions by direct manipulation, the system will indicate alternate ways that their goal could have been achieved using speech and/or pen.” Here is an example from his group’s map-based planning application.

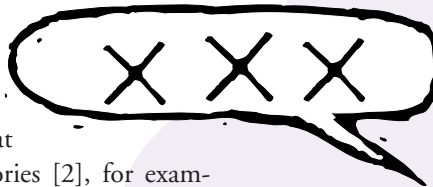


User: (selects “Info” button to display properties for a selected hotel)

Comp: Display information about the Hilton.



In another example, the user might use the mouse or pen to filter a hotel database based on attributes such as “pool,” “pets,” or “fireplace.”



User: (clicks on “pool” and selects the filter option)

Comp: I only want hotels with a pool.

This technique of educating users with verbal feedback while they’re carrying out tasks has the advantage of providing concrete examples of acceptable utterances in a context that the user is likely to understand and remember.

Summing Up

Letting users know what to say in a speech interface is challenging, whether or not a graphical interface is present. Here’s a summary of the prompt design techniques in use in today’s speech-only systems that help users formulate acceptable speech input:

- Explicit prompts help to constrain user input when only a limited vocabulary is available.
- Implicit prompts incorporating discourse strategies help shape user input in conversational systems.


PERMISSION TO COPY WITHOUT FEE, ALL OR PART OF THIS MATERIAL IS GRANTED PROVIDED THAT THE COPIES ARE NOT MADE OR DISTRIBUTED FOR DIRECT COMMERCIAL ADVANTAGE, THE ACM COPYRIGHT NOTICE AND THE TITLE OF THE PUBLICATION AND ITS DATE APPEAR, AND NOTICE IS GIVEN THAT COPYING IS BY PERMISSION OF THE ASSOCIATION FOR COMPUTING MACHINERY. TO COPY OTHERWISE, OR PUBLISH, REQUIRES A FEE AND/OR SPECIFIC PERMISSION. ©ACM 1072-5520/96/1100 \$3.50

- Incremental or expanded prompts are fast when user input is well formed and provide help when it is not.
- Tapering helps instruct users the first time they encounter a specific prompt and then speeds interaction when the same activity is encountered again.
- Hints help guide inexperienced users and speed interaction as they gain experience.

And in multimodal systems:

- A simple convention, “say what you see,” can guide user input.
- The display can be used to suggest or explicitly indicate what users can say.
- Speech output corresponding to users’ actions can provide contextual examples of spoken input.

Although these techniques can be effective, there is no substitute for user experience. Either through training or by exposure to an interface, the more users interact with a system, the more likely they are to know what to say. Patti Price of SRI reports that “some of our experiments have shown that the second 10 minutes of interaction have about half the error rate of the first 10 minutes of interaction.” Not only do users learn the allowable vocabulary and grammar of a system, but they tend to adapt their manner of speaking in a way that the computer is more likely to understand. For example, users learn to not speak before the computer is ready to listen and they learn to not pause in the middle of a sentence (which causes the computer to process only what was said before the pause).

User experience coupled with thoughtful prompt design does not guarantee a successful speech interface. Other design issues play an equally important role. These include crafting grammars or command languages, providing appropriate feedback, and recovering from errors. In addition, a consistent “sound and feel” across speech applications can go a long way in helping users know what to say. 

References

1. Gamm, Stephan, Haeb-Umbach, R., and Langmann, D. The Usability Engineering of a Voice Controlled Answering Machine. Presented at 15th International Symposium on Human Factors in Telecommunications, Melbourne, 1995.
2. Gamm, Stephan, and Haeb-Umbach, R. User Interface Design of Voice Controlled Consumer Electronics. *Philips Journal of Research* 49, 4 (1995).
3. Kamm, Candance. User Interfaces for Voice Applications. In *Voice Communication Between Humans and Machines*. National Academy Press, Washington, D.C., 1994.
4. Yankelovich, Nicole, Levow, Gina-Anne, and Marx, Matt. Designing SpeechActs: Issues in Speech User Interfaces. *Proceedings of CHI '95, Conference on Human Factors in Computing Systems*, Denver, CO, May 7–11, 1995.