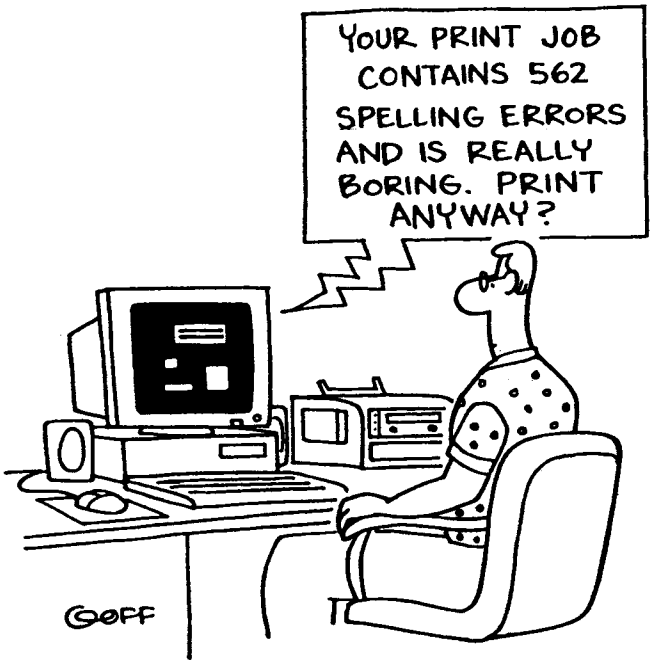
Chapter

10

Speech Guidelines



From:
Weinschenk + Barker.
Designing Effective
Speech Interfaces.
ISBN 0-471-37545-4

"We think in generalities, but we live in details."
Alfred North Whitehead

In the previous chapter on the laws of interface design, we covered the basic principles of interface design. This chapter applies those laws and provides specific guidelines to follow when designing the interface for a speech application. The guidelines are organized into the following categories:

Errors;

Feedback;

Confirmations;

User expectations;

Keypads and motor actions;

Social and environmental issues;

Command-and-control;

Continuous speech recognition;

Conversation and prompting;

Menus;

Non-speech audio/auditory icons.

Errors

Being error-prone, current speech technology is very much a matter of dealing with errors. One of the roles of the interface is to help reduce the number and severity of the errors the user can make, as well as to help mitigate the effects of the errors the computer can make.

Use Specific Error Messages

Error messages should be specific and tell the user exactly what to say or do. An inadequate error message either gives no information ("An error has occurred") or merely repeats a command. Consider the following exchange between a computer and a user:

System: "Say the departure date."

User: "Tomorrow."

System: "Say the departure date."

User: "I want to travel tomorrow."

System: "Say the departure date."

The user does not know what response the system wants and the system is not providing helpful or new information. An adequate error message tells the user what is wrong, why, and how to correct it. An example of this is shown in the following improved exchange:

An adequate error message tells the user what is wrong and why, and how to correct it.

System: "Say the departure date."

User: "Tomorrow."

System: "I do not understand that date. Say the month, date, and year. For example, say October 13th, 1999."

User: "July 1st, 1999."

In this example the system indicated that it did not understand the user's initial response and provided a specific format for a reply.

Limit Background Noise

Background noise leads to an increase in insertion errors. The computer thinks the user has said something, when what it really heard was background noise. Minimize background noise as much as possible.

Allow the User to Turn Off the Input Device

Make sure it is easy for the user to turn off the input device, such as a microphone. Users may need to have a different conversation with another person or machine, need to think about their next step, or gather materials or information for the next step. Allowing the user to turn off the input device eliminates background noise and, thus, reduces insertion errors.

If the interface is multi-modal, provide an icon, such as a picture of a microphone, to allow the user to turn off the input device. Figures 10.1 and 10.2 show examples from IBM's ViaVoice. The icon toggles on and off when clicked.

Figure 10.1 Microphone icon turned on in IBM's ViaVoice.

Figure 10.2 Microphone turned off in IBM's ViaVoice.

If there is a physical device, such as a microphone, allow the user to mute it with a button press on the device itself. If there is no visual or physical device, then allow the user to issue a command that turns off the input mode. For instance, in some applications you can put the speech interface into "sleep mode" and then issue a "wake" command to re-activate input mode.

Provide an Undo Capability

Because limitations in the technology mean that there will be errors, make it easy for users to undo previous actions. Build in ways for users to cancel out, go back, and undo actions.

Build in ways for users to cancel out, go back, and undo actions.

Use an Auditory Icon

Play an auditory icon to signal when an error has been made or when an error message is about to be shown or spoken.

Use Multi-Modal Cues for Errors If Appropriate

If the interface is multi-modal, use more than one mode to signal an error. For example, use an auditory icon and an error message dialog box if there is a visual/GUI component to the interface. Then speak the error message feedback while the error message dialog displays on the visual interface.

Consider Offering Replay

Replay modes allow the user to replay a particular command, menu, or prompt. They can help, but are not used very often by users.

Don't Assume People Hear Everything

You can't assume that people heard something just because the system spoke it to them. Put important information first or last to improve the likelihood of its being heard and remembered.

Feedback

People need feedback from the computer to know what is going on during an interaction. When a user issues a command, they want the system to acknowledge that they have been heard. They also want feedback when the system is busy so they know when to wait.

Supply Alternative Guesses

Although technology continues to improve, there are still limitations that result in errors. For example, a user may say one word, but the computer hears a different word. One way to help with these corrections is for the computer to provide the user with alternative guesses. For example, if the user says "Boston" and the computer is unsure of what was said, it could respond, "Did you say Austin or Boston?"

Acknowledge the User's Speech

After users speak, they expect the computer to respond with appropriate feedback. They want to be sure that the computer has heard what they have said. If the user has requested that an action be taken, the best type of response is to carry out the action itself. For example, if the user has spoken "2" to go to menu option two, then the best form of feedback is for the system to go on to menu option two.

Show When It Is the User's Turn to Talk

People need feedback to know whether it is their turn to speak or the computer's, and to know what is going on. If the interface is multi-modal, use visual feedback to indicate status or whose turn it is. If the interface has no visual component, use auditory cues to signal when it is the user's turn. Figure 10.3 shows this feedback for ConversaWeb software.

Figure 10.3 ConversaWeb signaling it's the user's turn.

Allow for Verification

People tend to verify more when using a speech interface than a visual interface. Make it easy for users to verify and check the status of what is happening now and what has happened recently.

Use Visual Feedback

If you have a visual component to your interface, use visual feedback. This reduces the amount of verification the user will have to ask for. Since the visual feedback continues to display, the user does not have to remember what was said.

Use Non-Speech Audio

In a paper by Lisa Stifelman et al. (1993), researchers talked about the use of speech feedback and non-speech audio. For example, in an early design of a voice recording design, when the user wanted to move from one category to another, they would speak the category name. The system would repeat the category name as a cue that it had moved to that category:

User: "Things to Do."

System: "Things to Do."

This was not effective feedback, however, because it was not clear to the user whether the recorder had actually moved to the new category or was just repeating what was said for clarity and confirmation. In the next iteration, the researchers added the phrase "Moving into" to signal the move:

User: "Things to Do."

System: "Moving into Things to Do."

This feedback conveyed that the system had moved into the new category, but now the researchers found the feedback was too long and wordy. Their next tactic, therefore, was to try a non-speech feedback. They used a short sound effect (called an *auditory icon* or *earcon*) to signal that the move had been made.

User: "Things to Do."

System: Makes an earcon sound.

Users rated their satisfaction higher with the earcon than with the other feedback methods.

Avoid non-speech feedback that sounds like equipment noise.

Avoid non-speech feedback that sounds like equipment noise. For example, don't use static as an auditory icon.

Use In-Progress Messages

If there is more than a three second-delay between when the user issues a command and when the system responds, issue an in-progress message immediately and repeat it every seven to ten seconds until the system responds (Gardiner-Bonneau, 1999). A delay of two to three seconds

is acceptable. An auditory icon during the delay (such as music or a tone) does not increase user acceptance of the delay. Here is a sample dialog between a user and a computer properly using an in-progress message:

A delay of two to three seconds is acceptable.

User: "List the categories."

System after two seconds: "Retrieving the categories. Please wait."

System after seven to ten seconds: "Your request for categories is still being processed. Please wait."

Confirmations

Confirmations are questions you ask of the user to be sure that the user has been heard correctly. We discuss here guidelines for using effective confirmations.

Use Confirmations Appropriately

Since there are many errors associated with the computer recognizing human speech, you may have to use confirmation questions to assure that the computer has heard the right phrase or message. Here is an example:

System: "What do you want to do next?"

User: "I want to schedule an appointment with my manager."

System: "Do you want to set up an appointment?"

User: "Yes."

You have to balance the cost of making an error with the extra time and annoyance in requiring the user to confirm a lot of statements.

Ask for Clarifying Information

If the computer cannot figure out what the user wants to do, have it ask a clarifying question. For example, consider the dialog in the previous section. If the user's final response has more than one possible meaning, the computer could ask, "Do you want to set up an appointment or contact the person by phone?"

Use Confirmations for Destructive Actions

If the user's action is destructive, require a confirmation.

If the user's action is destructive or will delete large amounts of information, require a confirmation. However, you can minimize the use of confirmations with expert users.

Be Specific

If the system does not recognize what the user has spoken, be specific about what the system needs. Use "Please repeat the date again," rather than, "Please repeat." Do not use a confirmation such as, "Do you mean October 19?" unless you are fairly sure you know what the user said.

User Expectations

One of the biggest challenges in the design of speech interfaces is dealing with user expectations. Users expect talking computers to understand more than they are capable of, and to provide more information than they are capable of. Your interface design can help set appropriate expectations.

Identify the Computer

Users will adjust their speech and their expectations if they know they are interacting with a computer and not with a human. They will speak more briefly and exactly, and enunciate more clearly. Let the users know up front that they are speaking to a computer. For example, start an interaction with "Welcome to the XYZ Voice Response System. What directory to you want to search for?" rather than "What directory do you want to search for?"

Let the users know up front that they are speaking to a computer.

Use enough words and a long enough greeting to convey this information. In a study by Susan Boyce (1999), she used different versions of a greeting. The first three were recorded human voices:

> "AT&T. How may I help you?" (Standard version)
>
> "AT&T Automated Customer Service. How may I help you?" (Short automated version)
>
> "AT&T Automated Customer Service. This service listens to your speech and sends your call to the appropriate operator. How may I help you?" (Long automated version)

In a fourth version, the long automated version was used but it was played with a computer-synthesized voice rather than a human voice. Both longer versions resulted in shorter utterances from the people calling in. The short automated version did not help shorten the utterances—people did not seem to catch the automated connotation. The shortest utterances resulted from the synthesized voice, but that version had the lowest satisfaction rating.

In a follow-up study Boyce used an audio logo to signal an automated system, and then used a brief "How may I help you?" This proved as effective as the longer greetings. Guidelines on initial greetings, therefore, are the following:

- Use a longer greeting or an audio cue.
- Explicitly state that it is an automated system.
- If possible, use a recorded human voice rather than a synthesized voice.

Build in Training Time

Most people have not used speech interfaces, and will need time to get used to talking to the computer. Just as when GUIs first became available and people needed training on mice and windowing, first-time users of speech interfaces need to get used to this new way to interact.

Introduce the Voice

Provide an introduction or training message to familiarize the user with the system's voice.

Keypads and Motor Actions

If the interface includes a keypad or a motor component, such as a keyboard or mouse, here are some guidelines for these devices.

Avoid Key Combinations

If the interface includes keypads, avoid key combinations. For example, do not require the user to press the # key after each command or the * key before each command. These add time and keystrokes to the task.

Use ISO Standard Letter Mappings

If you are using a keypad, for instance with telephone interactive voice response systems, use the ISO standard letter mappings as shown in Table 10.1

1	None
2	A, B, C
3	D, E, F
4	G, H, I
5	J, K, L
6	M, N, O
7	P, Q, R, S
8	T, U, V
9	W, X, Y, Z

Table 10.1 ISO Standard Letter Mappings

Q and Z are not printed on most telephone keypads. If you use these, you must remind the user where to find them.

Reserve 1 and 2 for Frequent Actions

Reserve the numbers 1 and 2 for frequent and critical actions. Do not use these numbers for destructive commands, such as *delete*.

Avoid numbers 5 and 8 for destructive commands. Since the numbers 5 and 8 on a keypad are usually surrounded on all sides with other numbers, they are the most likely to be pressed accidentally.

Use the Appropriate Word for Keypresses

Use the word *enter* when you want the user to press multiple keys on a keypad. Use *press* when you want the user to press one key.

Assign Command Labels Consistently

Be consistent with commands. Assign the same key or word to the same function across the entire application.

Social and Environmental Issues

Interfaces are used in the context of people's work and their physical environment. The guidelines here relate to these social and environmental issues.

Decide on Flexibility

You will need to decide on how flexible your interface is for different users and tasks. Maximum flexibility is not always best. If users are under stress and need to do some tasks quickly or infrequently, it may be best to lead them through step by step without a lot of flexibility. If, on the other hand, they need to complete simple tasks quickly and in an unpredictable order, you may want to build in flexibility. Flexibility is a continuum with no particular right or wrong point.

Flexibility is a continuum with no particular right or wrong point.

Consider Stress

People react to all stimuli differently when they are under stress. For example, an easy application can become difficult if the user is also dealing with a customer or

co-worker who is upset. Analyze the amount of stress that will be in the environment when people are using your interface. Then analyze the amount of information and the type of activities your interface requires. For users under stress, you may need to make design decisions (such as reducing the amount of information or the amount of navigation) to compensate for the confusing effect of stress.

Consider Social Interaction

If people use software while they are interacting with another person, you must be aware of the impact your interface has on the social interaction between the individuals. For example, a salesperson speaking with a customer does not want to be distracted or have the customer distracted by the computer. If the interface becomes a distraction to the human interaction, its effect will be negative. Consider carefully how introducing a computer into a social interaction may affect the interaction as a whole.

Be aware of the impact your interface has on the social interaction.

Match the User's Work

Make sure you know how the user is going to do the work when the new software is in place, and design for that order and process of work. Don't make users change an optimal work flow to fit your design.

Don't make users change an optimal work flow to fit your design.

Command-and-Control

Command-and-control speech applications refer to a type of speech technology that recognizes a limited vocabulary of individual words and phrases spoken by a user. There are some special guidelines to follow when designing an interface for command-and-control applications.

Don't Use "I"

In a command-and-control interface, the system recognizes only certain commands and terms spoken by the human. If the computer uses the first person, "I," users tend to talk in longer phrases, make more vague requests, and are more polite. These tendencies lead to more errors in systems that require specific commands.

Use Modeling

In a GUI interface, users are constrained as to their behavior—they can press only certain buttons or controls. In a speech interface, however, users can say anything (although the system will not always respond well). To constrain user speech to a form that is better understood and minimizes errors, use modeling to convey to the user the terminology and structure they should use. For example, this first script does not model constrained speech and results in unconstrained speech by the user:

> System: "Welcome to ABC Travel Automated System. We look forward to servicing your travel needs. What are the dates of travel that you would like me to check for?"
>
> User: "We are interested in traveling the first week of July, say July 1st to July 5th."

If the speech engine can handle this type of continuous speech, then you will not have a problem. But in a command-and-control system, you require more exact and constrained speech. In this case the beginning statement by the system is too open—it implies the system can handle continuous, natural language.

If you need to constrain speech, give the user a role model:

> System: "Welcome to ABC Travel Automated System. Say the departure date of travel. For example, say October 1st, 1999."
>
> User: "July 1st, 1999.
>
> System: "Thank you. Say the return date of travel."

In this model, the user will follow the lead of the computer and speak in the same manner, resulting in fewer errors.

Be Brief and Terse

People model the length of system speech. If the computer speaks in short, terse sentences, the users will also.

People model the length of system speech.

Continuous Speech Recognition

Continuous speech recognition systems allow the user to talk to the system without stops and pauses. Continuous speech recognition systems can recognize more utterances than a command-and-control system. The guidelines for continuous speech recognition differ somewhat from those for command-and-control.

Use "I"

In a continuous speech system, humans can speak in a continuous stream. Susan Boyce's research (1999) shows that when the system speaks using the first person, "I," users report higher satisfaction with the system. The use of "I" in a continuous recognition system does not degrade user performance.

When the system speaks using the first person, "I," users report higher satisfaction with the system.

Simulate a Conversation

When two people converse, they take turns talking and listening. A speech interface should do the same. When the computer finishes speaking, it should pause and wait for the human to respond.

Avoid Modal Interference

If the user is speaking to another person, for instance, a customer, and is then required to also speak to the computer, this produces modal interference. Modal interference will cause the user to be distracted and confused.

Conversation and Prompting

What screen design is to graphical user interface design, conversation and prompting are to speech interface design. Taking care when designing the conversation and prompting will reduce errors and increase user satisfaction.

Choose the Appropriate Word

Use the word "say" when you want the user to speak, rather than "enter." Use "enter" for key presses. If you want the user to speak the words "yes" or "no," then use the phrase, "Say yes or no." Here is an example:

> "Do you want to transfer funds now? Say yes or no."

rather than:

> "Do you want to transfer funds? Enter yes or no."

Use as few words as possible between "say" and the command. This gives the user less to remember. Since speech interfaces place a heavy memory demand on users, it's important to be as brief as possible and still communicate clearly.

For example use: "Say yes or no."

rather than:

"Say the word yes or no."

Avoid Personal Pronouns when Asking for a Response

Avoid using a personal pronoun when asking the user to respond to a question. For example, use:

"Say your credit card number."

rather than:

"Tell me your credit card number."

Using a personal pronoun places an extra word between the command (say) and the information the user is supposed to respond with (credit card number). To decrease the memory demand, use as few words as possible between the two elements. In addition, use the command "say" as it is clearer and more imperative than "tell."

Change Voices Appropriately

Different voices can be used to signal a mode change, for instance, when the user is in one part of the application versus another. Different voices can also be used for different languages. If you do not want to signal a different

mode or language, then use the same voice throughout. Warnings, however, should be presented in a voice that is qualitatively different from other voices used by the system so the user will be alerted to pay attention.

Decide on Alerting Tones

If synthesized speech is used exclusively for warnings, don't use alerting tones.

If synthesized speech is not used exclusively for warnings, then use an alerting tone to precede warnings.

Use Small Steps

In a visual interface, people can accomplish fairly large, complex tasks without specific instructions. Speech interfaces, however, strain the user's memory capabilities, so they need to work in smaller steps. Unlike GUIs, speech interfaces users prefer the computer to query them at each step.

Avoid Long Prompts or Menus

If there is no visual element to the interface, you should avoid using prompts or menus with many options. Human short-term memory quickly reaches its limit in speech interfaces. Users are able to remember about three or four menu or prompt options at a time.

Avoid Jargon and Technical Terms in Prompts

Make sure that the users will understand all the terminology used in your prompts. Avoid jargon and technical terms in prompts.

Use Progressive Prompting

When you need to prompt the user, start with short high-level prompts such as:

> System: "Welcome to the XYZ system. What would you like to do?"

If the user does not respond or gives an unrecognizable response, then go to:

> System: "You can check your account status, obtain current market quotes, or view a list of other actions."

If the user still does not respond or gives an unrecognizable response, then go to:

> System: "Say one of the following: Account Status, Quotes, or List."

Use Prompts to Signal Going Back

If there are no visual cues or actions to take the user back to a previous step, use a speech cue, such as "What Next?" This tells the user the dialog has stopped, and the system has returned to a previous node.

Make Prompts Directive and Exact

In a discrete system (as opposed to a continuous speech system) where a user must speak a specific phrase or word, using synonyms can result in an incorrect command by the user. For example:

> System: "Say Activate Recording to turn on the recording feature."
>
> User: "Turn on Recording."

The inexact and inconsistent wording of the prompt (*activate* versus *turn on*) influenced the speaker to use a synonym. Use this wording instead:

System: "Say Activate Recording to activate recording."

User: "Activate Recording."

Use an Appropriate Time-Out Period and Action

If the system has requested a response from the user and the user has not responded, use the time-out periods and actions shown in Table 10.2. In general, use ten seconds as the time-out period unless the user has to compose or look up information. The type of system action depends on the user task.

Allow Non-GUI Terms

Spoken language is less formal than GUI terms on a window.

Users recognize GUI terminology on a computer screen, but do not necessarily expect those words to be spoken. Nor do they expect they should have to speak them. For instance, *delete* is a common word on a button in a GUI interface, but users may say "remove" or "erase" instead. Spoken language is less formal than GUI terms on a window. Therefore, the computer should accept both GUI and non-GUI terms.

Allow Relative Dates

Users tend to refer to dates in a relative way.

In a GUI interface, it is common to see dates such as 09/03/1999. In speech, however, users tend to refer to dates in a more relative way, such as "next Friday," or "the day after tomorrow." Although some of these may be hard to interpret, try to interpret them as best you can.

Say a standard command	10 seconds	Repeat prompt. If no response, give message and connect to operator if available
Say continuous speech that may require composition or look-up	Unlimited time-out	None
Say a menu choice	10 seconds	Repeat menu. If no response, give message and connect to operator if available
Press a key or use a pointing device	10 seconds	Repeat prompt. If no response, give message and connect to operator if available

Table 10.2 Time-Outs and System Actions

Avoid Long Pauses

People are uncomfortable with long pauses in conversation. If you have long periods of silence, users will want to fill that space. They will, therefore, talk more, using less meaningful words and sounds, for instance, uhs or umms. This will result in more errors. Use an "in progress" message or a non-speech sound when a long pause is necessary.

Choose an Appropriate Speed

Users will mimic the speed of the computer.

Users will mimic the speed of the computer. If the computer is reacting and speaking slowly, the user will tend to slow down. If the computer is reacting or speaking quickly, the user will speed up. Select an appropriate speed that keeps the interaction moving naturally, but minimizes errors. Use an appropriate speech rate: for natural speech, use a rate between 150 to 180 words per minute; for synthesized speech, use 150 to 170 words per minute.

Use "Barge-In" Techniques

Allow users to interrupt the computer. If there is a visual component to the interface, they can click on a button to interrupt what the computer is doing or saying. If there is no visual component, barging in refers to the user speaking to stop the computer.

Use Tapering

If you must provide the same message or information over and over, start with more detailed instructions and then taper the number of words used. For example:

> System: "Say the street address, city, and state of the business you are interested in."

The next time, use less detail.

> System: "Say the street, city, and state."

Eventually use the least amount of detail.

> System: "Say the business address."

This mimics the way people converse. If you do not use tapering, the conversation is slower and users will become impatient.

Be Consistent

Use the same terms in the documentation as you do in the interface.

Use Prosodic Features

Prosodic features are pitch, volume, speed, and pause. You can vary the pitch of spoken text to introduce new topics and emphasize important sections of work. Increasing pitch also makes the computer sound more lively. Varying speech too much can make the output hard to hear, so use caution. People are used to a human-like intonation of sounds and words which computers cannot easily simulate. Try to get the prosody as natural as possible.

Increasing pitch makes the computer sound more lively.

You can vary the volume in the same way as pitch, but not as much. Volume quickly becomes either too loud or too soft.

Slowing down speech makes it easier to understand, but can be annoying if it is too slow.

Pauses are important in human-to-human conversation, but they can be ambiguous in human-computer interaction. Use pauses to suggest that it is the user's turn to act.

Menus

Some speech-only systems, especially IVR (interactive voice response) systems, rely on menus as their main navigation tool. The design of these menus is therefore critical to the usability of the system.

Use Result-Action for Menus with More Than Two Options

In a menu with more than two options, describe the result first and then the specific action. For example:

> "For a list of categories, say 1. For a list of people, say 2. To access your account status, say 3. To open a new account, say 4."

rather than:

> "Say 1 for a list of categories. Say 2 for a list of people. Say 3 to access your account status. Say 4 to open a new account."

If you state the menu choice first, the user may forget which choice they want before hearing all the menu options.

Use Action-Result for Menus with Two Options

If there are only two options, place the action first, as in:

> "Press 1 for yes, or 2 for no."

Let the User Know When the Menu Is Complete

Cue the user that all the menu choices have been given. You can do this by telling them to select one—for example, "Make your selection now"—or by the inflection in the voice.

Order the Menu Appropriately

Place the most frequently-used options at the top of the menu hierarchy, both in terms of nesting as well as within a

particular menu. If a menu item is temporarily unavailable or unavailable to that user, leave it off the menu.

Use Results as Feedback

The best feedback for a menu option is to present the results of the choice as shown in the dialog here:

System: "For a list of categories, say 1."

User: "1."

System: "The categories are..."

Do not use a confirmation message for menus. For example, do not use:

System: "For a list of categories, say 1."

User: "1."

System: "You chose 1. The categories are ..."

Constantly telling users what they chose is redundant and takes time. Users will grow tired of the extra wait.

Use an Appropriate Amount of Nesting

If the user does not know what they want until they hear the options, then limit the number of menu options to four and use nesting (sub-menus). This way, users will choose a category first, and then go to the choices for that category. If, however, they know what item they want, then breaking up the choices into categories will waste time. Minimize nesting in this case, and allow more than four items per menu.

Non-Speech Audio/ Auditory Icons

An auditory icon is a non-speech audio signal based on a real-world sound. For example, the sound of a police siren signaling an emergency is an auditory icon. You can alter the character of the sound to imply a change in the characteristic of the object. For example, a cash register sound can get louder the more money is deposited.

Use Standard Sounds with Their Usual Meanings

Many auditory icons are already used in other applications. If you are going to use these, be sure to use them in an expected context. Table 10.3 lists some common auditory icons and their uses.

Busy signal	Action is not available because it's already being used by another
Beep	Error or attention
Siren	Emergency
Dial tone	Open line, ready

Table 10.3 Auditory Icons and Their Meanings

Stay within an Octave for a Sequence

If you are using a sequence of tones that must be distinguished from another sequence, keep each sequence within an octave so that you can use the octave change as a signal of a different sequence.

Use appropriate scale movements. For example, rising tones imply "up" and falling tones imply "down."

Rising tones imply "up."

Use Volume to Signify Meaning

A tone that grows quieter indicates that something is going down. A tone that grows louder signifies that something is going up. Avoid very loud noises unless you purposely need to startle users.

Interview with Sharon Oviatt

Co-Director, Center for Human-Computer Communication

Department of Computer Science and Engineering

Oregon Graduate Institute of Science and Technology

Sharon Oviatt is co-director of the Center for Human-Computer Communication (CHCC) in the Department of Computer Science and Engineering at Oregon Graduate Institute of Science and Technology. Dr. Oviatt has a Ph.D. in Experimental Psychology from the University of Toronto (1979), an M.A. from the University of Toronto (1974), and a B.A. (with highest honors) from Oberlin College (1972).

Oviatt has researched many of the detail decisions we make on speech interfaces. Her discussion about these decisions, the guidelines, and their implications is presented in the following interview.