Universal Speech Interfaces

Ronald Rosenfeld, Dan Olsen and Alex Rudnicky

In recent years speech recognition has become commercially viable on off-the-shelf computers—a goal that has long been sought by both the research community and by prospective users. Anyone who has used speech recognition technology understands that it has many flaws and much remains to be done. Uncertainty exists about how speech can and should be used, as well as how recognition algorithms and tools for speech-based applications should be developed. Achieving reliable, accurate speech recognition is similar to building an inexpensive mouse and keyboard. The underlying input technology is available, but the question of how to build the application interface remains. We have been considering these problems for some time [8]. In this paper we present some of our thoughts about the future of speech-based interaction. This is not a report of results we have obtained, but rather a vision of a future to be explored.

Speech Recognition

From the inception of speech recognition research, achieving "natural" interaction with computers has been cited as the primary benefit. The concept of talking with a machine as fluently and comfortably as with another human being has attracted funding and interest. We wanted to create HAL from *2001: A Space Odyssey* with a gentler personality. Naturalness of communication, however, is not the only goal of speech recognition, nor is naturalness reserved for speech. Based on the idea that "a picture is worth a thousand words," graphical user interfaces (GUI) have been developed purposefully for their naturalness. If naturalness is not the key driver for speech, what is? We think there are at least three fundamental advantages for speech.

1. **Speech is an ambient medium rather than an attentional one**. Visual activity requires our focused attention, whereas speech allows us to interact while using our other faculties (e.g., visual, sensory-motor) to do something else.
2. **Speech is descriptive rather than referential.** When we speak, we describe objects by their roles and attributes. In visual situations we point to or grasp objects of interest. For this reason, speech and pointing are to a large extent complementary and can often successfully be combined.
3. **Speech requires modest physical resources.** Speech-based interaction can be scaled down to much smaller and much cheaper form factors than can visual or manual modalities.

We see a future for speech, not so much for its naturalness but for its ubiquity. Natural-language interfaces are still an interesting and important topic, but speech interaction is both more and less than that. For this reason we choose to focus on speech as an input-out-

Ronald Rosenfeld,
School of Computer Science
Carnegie Mellon University
roni@cs.cmu.edu
Dan Olsen,
Brigham Young University
dolsen@acm.org
Alex Rudnicky,
School of Computer Science
Carnegie Mellon University
air@cs.cmu.edu

put modality rather than as a medium for natural language. We see interactive systems as mechanisms for humans to express needs to and obtain services from machines. By machines we mean not only computers in the popular sense, but also any gadget, appliance, or automated service that, in order to be fully used, must be reconfigured, controlled, queried, or otherwise communicated with. We are surrounded by dozens of such machines today. The exponential drop in the fundamental cost of computing will cause hundreds more to be developed in the near future. Examples of such interactivity include:

- Configuring and using home appliances (videocassette recorders, microwave and convection ovens, radios, alarms);
- Configuring and using office machines (fax machines, copiers, telephones);
- Retrieving public information (e.g., weather, news, flight schedule, stock quotes);
- Retrieving and manipulating private information (e.g., bank or other accounts, personal scheduler, contact manager, other private databases);
- Handling asynchronous communication (voice, e-mail, fax); and
- Controlling miscellaneous user and consumer applications (map following, form filling, Web navigation).

To clarify our ideas, we distinguish between intelligent machines and simple machines. Suppose that you had a household robot like that envisioned by Isaac Asimov. While seated at the table, you might hand a dirty dish to the robot and say, "Take care of this." This is a natural statement that most children would understand. The desire is to have the dish taken to the kitchen, rinsed, and placed in the dishwasher. The naturalness of this communication derives from the large amount of world knowledge that the robot must possess to infer the correct response. This is not so much an interaction problem as it is a natural language and inferential reasoning problem. This problem is not necessarily related to speech. A more natural form of communication would be to hand the dirty dish to the robot and then expect the right thing to happen without any speech at all. We would consider this an intelligent machine.

Only speech will scale as digital technology progresses.

For speech interfaces we focus on simple machines. With a simple machine the user can, at least in principle, get a mental model of the machine's capabilities and of the machine's rough state. Moreover, the user is assumed to already know what is desired, although he or she does not have to know how to get it done. Under this paradigm, high-level intelligent problem solving is performed by the human; the machine is only a tool for getting needed information, modifying it, or issuing instructions to the desired service. We find speech interaction with simple machines to be interesting because it sidesteps the artificial intelligence problems of natural language in favor of the ubiquitous nature of speech interaction. We believe that in the future human beings will be surrounded by hundreds if not thousands of simple machines with which they will want to interact.

In particular, the approach we are proposing is not aimed at applications requiring truly intelligent communication. For example, an air travel reservation system will fall within our focus only if the machine is used to consult flight schedules and fares and to book flights and the user plans and makes decisions. The machine acts as a passive travel agent that does not do much thinking on its own but mostly carries out the explicit requests of the user. The more intelligent travel agent, however desirable, is outside the scope of this discussion.

Why Speech?

In dealing with simple machines you might ask why you should go to the effort of recognizing speech when a visual technique would suffice. When considering technologies that might meet the needs of ubiquitous interactivity, you should include situation, cost, breadth of application, and the physical capabilities of human beings. When we talk of embedding interactivity into a wide variety of devices and situations, the keyboard and mouse are not acceptable interactive devices. When a user is not seated at a flat surface, a keyboard or mouse will not work. In any application with richness of information, a bank of buttons is unacceptably restrictive. In a large number of cases only speech provides information-rich interaction and meets the form factor requirements of the physical situation.

If we consider future exponential growth, it is clear that any interactive solution relying primarily on processing and memory capacity will over time become very small and very inexpensive. Speech interaction requires only audio input-output devices (namely, a microphone and a speaker), which are already quite small and inexpensive, coupled with significant processing power, which is expected to become inexpensive. No such projections hold for keyboards, buttons, and screens. Visual displays have made only modest gains in pixels per dollar over the last 20 years, and no significant breakthroughs are expected. Visual displays are also hampered by the size requirements for discernible images and by the power required to generate sufficient light energy. Buttons are cheap but are restricted by size and range of expression. Any richness of interaction through the fingers quickly becomes too large and too expensive for ubiquitous use. Only speech will scale as digital technology progresses. A personal digital assistant can get more powerful, but it cannot get physically smaller. Its size and form factor are dominated by its screen, touch pad and the limitations of human beings [5]. Speech interfaces have much smaller minimal sizes and power requirements.

Spoken language dominates the set of human faculties for information-rich expression. Typical human beings, without a great deal of training, can express themselves in a wide variety of domains. As a technology for expression, speech works for a much wider range of people than typing, drawing, or gesture because it is a natural part of human existence. This breadth of application is important to ubiquitous interactivity.

State of the Art in Developing Speech Interfaces

Speech interfaces are only beginning to make an impact on computer use and information access. The impact thus far has been limited to places where current technology, even with its limitations, provides an advantage over existing interfaces. One such example is telephone-based information-access systems, because they provide a high input bandwidth in an environment where the alternative input mode, DTMF (TouchTone telephone buttons), is inadequate. We believe that speech would achieve much higher adoption as an interface technology if certain fundamental limitations were addressed, particularly,

- Recognition performance
- Accessible language (for users)
- Ease of development (for implementers)

That is, it should be possible for users to verbally address any novel application or artifact they encounter and expect to quickly be involved in a constructive interaction. At the same time it should be possible to dramatically reduce the cost of implementing a speech interface to a new artifact so that the choice to add speech is not constrained by the cost of development.

We consider three current approaches to the creation of usable speech systems: natural language, dialog trees, and commands.

Natural Language

First, we can address the problem of accessible language by allowing the user to use unconstrained natural language in interacting with an application. That is, given that the user understands the capabilities of the application and understands the properties of the domain in which it operates, he or she is able to address the system in spontaneously formulated utterances. In this case, although the onus is removed from the user to learn the language supported by the application, the onus is instead placed on the developer. The developer needs both to collect a large corpus of user expressions and to create an interpreting component that maps user inputs into application actions. Examples in the literature of this approach include ATIS [7] and Jupiter [13].

This approach not only places a huge usability burden on the developer to understand any reasonable utterance by the user, but also carries the additional burden of supporting a discovery dialog. All computing systems and human beings have limitations on their knowledge and the services that they provide. For users to effectively use a system they must have a means for discovering those abilities and limitations. A structured language implicitly communicates the functional limits of the application: What is supported is exactly what can be expressed. Conversely, an unconstrained language must do so explicitly: With a natural-language approach the developer must not only develop the application language but also a sufficiently broad meta-language to support user discovery.

A natural-language approach attempts to exploit transfer of prior human experience with other people to simplify the learning of a new application. This handcrafted language development strategy, however, does not result in transfer of learning among human-machine interfaces. Even if learning does take place in the context of a particular application (say through the modeling of spoken utterance structure [12]), there is no expectation that this learning will transfer across applications, because development efforts are not systematically related.

A second problem with natural-language interfaces is that they do not systematically address the issue of constraining language with a view to improving recognition performance; the only improvement in recognition performance is through the accumulation of a domain-specific corpus. For the foreseeable future constraining possible alternatives is the most important method for producing high recognition accuracy. Even human beings in high stress or high noise situations use restricted formalized vocabularies for accurate communication. The freely spoken natural language model works against this need.

One of the primary contentions in favor of unconstrained natural language interaction is that there is zero learning time because all human experience transfers to an application that understands natural language. In practice there is no natural language experience that has zero learning time. All human-human situations require time to acquire terminology, shared history, shared goals and relationships of trust. Human beings regularly find themselves in situations where they do not understand and cannot communicate effectively even though all parties are speaking the same language. More importantly, unconstrained language does not inherently eliminate communication difficulties. Because the knowledge and capacities of human beings are so rich, we require rich language structures to build up the shared understandings that support natural language. Such rich mutual understandings, with their shades and nuances of meaning, are not only not required when communicating with "dumb machines" but in many cases an impediment to getting service. Simplifying the language and unifying its structure will reduce these problems. Simple machines as defined earlier are fundamentally limited in the services that they can perform. We want a system that will readily and efficiently communicate those limits so that users can make their own judgments about effective usage. We expect predictable, efficient subservience from our simple

Unconstrained language does not inherently eliminate communication difficulties.

machines, not independent behavior and thought.

As an alternative to allowing the user to speak freely (and compensating for this in the parsing and understanding component of the system) we can constrain what the user can say and exploit this constraint to improve system performance. We consider two such approaches, dialog-tree systems and command-and-control systems.

Dialog Trees

Dialog-tree systems reduce the complexity of recognition by breaking down activity in a domain into a sequence of choice points at which a user either selects from a set of alternatives or speaks a response to a specific prompt (such as for a name or a quantity). The drawbacks of such systems, from the user's perspective, center on the inability to directly access the parts of a domain that are of immediate interest and to otherwise short-circuit the sequence of interactions designed by the developer. A space with many alternatives necessarily requires the traversal of a many-layered dialog tree, because the number of choices at any one node will necessarily be restricted. From a designer's perspective such systems are difficult to build because they require being able to break down an activity into the form of a dialog graph; maintenance is difficult because it may require rebalancing the entire tree as new functions are incorporated. Although dialog-tree systems may be frustrating to use and difficult to maintain, they simplify the interaction as well as minimize the need for user training. What this means is that a user's contribution to the dialog is effectively channeled by the combination of directed prompts and the restricted range of responses that can be given at any one point. This is the approach commonly used in commercial development of public speech-based services.

Command and Control

Command-and-control interfaces reduce complexity by defining a rigid syntax that constrains possible inputs. The language consists of a set of fixed syntactic frames with substitution of variables (for example, "TURN VALVE <valve-id> TO POSITION <position-value>"). In a restricted domain, such a language provides the user with sufficient power to express all needed inputs. At the same time the recognition problem is simpli-

Speech interface approach:	Unconstrained Natural Language	Dialog trees	Command and Control
User's effort	low	moderate	high (moderate with use)
Developer's effort	very high	moderate	moderate
User training	none	none	required for each application
Supports discovery?	no	yes, inefficiently	possibly
Scales to complex tasks?	unknown	no	no
Scales to hundreds of applications?	yes, but effort not amortized	yes	no
Stress on speech recognizer & parser	high	low	moderate

Table 1 summarizes the properties of the three approaches to speech interfaces discussed above. As can be seen, no approach is satisfactory along all dimensions.

fied because the language is predictable and can be designed to avoid confusion. The utility of such interfaces depends on the willingness of users to spend time learning the language for such a system. The drawback of command-and-control systems is the investment that the user makes in learning the language. Being able to communicate with additional applications requires that this effort be duplicated. This may be feasible for a few applications but it is unlikely to apply to tens or hundreds of applications, which is what would be required of an interface that permits ubiquitous access to machines. This approach also suffers compared with dialog-graph-based systems in that this interaction style is not inherently self-explanatory; the user is not guided into the correct input style by the structure of the interface.

© Jose Ortega/Images.com, Inc.

Lessons from the GUI Revolution

In seeking a solution for effective development of spoken language interfaces we can learn from the history of GUIs. The current state of research on speech interaction and its supporting tools is similar to the state of research on GUIs in the early 1980s. At that time it was assumed that tools for constructing GUIs must remain general in their capabilities. A primary goal was to provide maximum flexibility for designers to create and implement any interactive dialog. It was clearly stated that tools should not introduce bias for the types of interactive dialogs that should be developed [2, 4].

The commercial introduction of the Apple Macintosh changed all of that thinking. The Macintosh clearly showed that wide variability in interactive dialog was harmful. A fundamental maxim of the Macintosh was its style guide and the toolkit that supported it. The Macintosh represented a shift away from all possible interactive styles to uniformity of interactive style. When menus, scrollbars, buttons, dragging, and double-clicking all work in the same way, global ease of use is much greater than when these items are uniquely crafted for each situation. Among the major keys to success of the Macintosh are that once a user learns the basic alphabet of interactive behaviors, those behaviors can be transferred to almost any Macintosh application, and when faced with a new Macintosh application the set of things to try is very clear. This transference of experience and the presentation of a clear space of exploration as a way to master new applications has been wildly successful in GUIs and has come to dominate virtually all interactive computing.

The uniformity of the Macintosh style also affects how interactive applications are developed. Creating tools that supported general dialogs of any sort was overshadowed by a widget-based strategy. In the widget strategy the paradigm is prebuilt, pretested pieces that are assembled to address the needs of a given application. These pieces are given parameters so that designers can fill in the application-specific information. The widget strategy for development has been extended to the notion of interactive frameworks. In such frameworks, developers are provided with a complete skeleton application with all standard behaviors predefined. The particular needs of the application are then fitted into this framework. This tool and implementation strategy not only simplified development but also reinforced the uniformity of applications. Both the cost of GUI development and the range of programmers with such skills improved dramatically.

The key insight of the Xerox Star and Macintosh revolution is that usability is a global rather than a local phenomenon. Users do not use one or two applications in the course of their work. They use many different applications and are repeatedly faced with learning new ones. Furthermore those applications must continually work together to create a productive whole. Uniformity in the basic interactive alphabet increases the transference of learning from one application to another and improves the overall usability of the entire interactive environment. Such transference becomes enormously important in a world of ubiquitous simple machines. A homeowner does not care nearly as much about saving a few seconds in effectively controlling the oven as she does about readily controlling every appliance in the house.

Speech interaction can learn further from experience with GUI standardization. When originally introduced, a standardized look and feel was presented as the key to learning transference between applications. Over time, however, we have learned that a uniform look (visual appearance) is not as important as a uniform feel (input behavior). Users readily adapt to scroll bars or buttons that have different stylistic appearances. They do not adapt as readily, however, to differences in interactive inputs. Uniform input behavior is important because the input behavior is invisible and must be remembered. This insight is critical in spoken-language interfaces because all interaction is invisible. Uniform structure, terminology, and input behavior are critical for an invisible interactive environment to be successful.

A single, universal metaphor should be used to cover a wide range of application classes.

Standardizing the "Sound and Say"

We have focused on simple rather than intelligent machines and thus escaped the difficulties of artificial intelligence. We have identified lessons from the history of graphical interaction to guide the movement forward. The next issue is to identify how spoken language interfaces to simple machines should be developed. Our solution is a universal speech interface (USI) style. By this we mean a standardized look and feel (or, rather, "say and sound") across varied applications and domains. There are several components to such a style.

- **Universal metaphor.** A metaphor allows the user to map concepts from a familiar conceptual landscape into a new, less familiar domain. The success of the GUI revolution depended to a great extent on the desktop metaphor. A metaphor for speech communication between human and machine is significant because the limited bandwidth of speech severely restricts the cues that can be provided at run time—the user must have a good mental model of the application and of how to accomplish various goals within it. Ideally, a single, universal metaphor should be used to cover a wide range of application classes. Any deviation from the metaphor will require substantial dialog between the human and the machine to identify the differences.
- **Universal user primitives**. Many aspects of dialog interaction are universal; that is, they recur in many applications. The list of such building blocks is quite long: detection and recovery of recognition errors, barge-in, taking or

relinquishing the floor, answering multiple-choice questions, asking for help, navigating, and so on. The user must have standard ways of expressing each of these. For a given application, idiosyncratic ways of expressing these primitives may be better. However, the uniformity, and the usability that uniformity brings, dominates other issues. When faced with a new spoken-language interface to a simple machine, the user should already know how to find out about that machine's capabilities. The interactive scaffolding that supports users must be uniform. Spoken-language studies by Arons [1] strongly indicate that language for orientation and navigation must be consistent.

- **Universal machine primitives.** Machine primitives are similar to user primitives in that a small set of machine prompt and response types recur in many applications and, in fact, constitute the great majority of turns in these applications. Standardizing these machine-side universals will allow the machine turns to be shorter and clearer. This could be accomplished by using appropriate common phrases, by making use of prosody, and by packing the audio channel with nonverbal information (beeps, melodies, and other "earcons"). [For literature on work in this area, see 3, 10.] If responses are uniform in style, user understanding is increased. Communicating with people is much more error-prone with speech than with visuals because of the transient nature of audio. The user has no opportunity to study audio output or to control its order of presentation. People expect to visually scan a display several times to understand its content. They are much less patient when an audio help message is played for the third or fifth time. Uniformity of the output simplifies the user's extraction of what she needs to know.

The use of a universal interface will greatly reduce the development time of spoken-language interfaces. Much of the interaction can be standardized and included in an application framework. This limits the difficulty of producing new speech applications. To the extent that we reduce costs and shorten the time required to develop a new application, we foster the creation of thousands of speech applications rather than tens.

The use of a universal interface simplifies the problem of speech recognition.

The use of the universal interface also simplifies the problem of speech recognition. The uniform treatment of universal primitives will result in a substantially lower expected branch-out factor for the user's language, with a direct impact on recognition accuracy.

We believe that such a USI is possible and will be effective precisely because of the restricted nature of computing technology. Devices and applications function by their information state. Communicating that state to the user and allowing that user to modify the state are fundamental to such interactive situations. Any interaction style that does this clearly, effectively, and uniformly will to some level be successful as a USI. Virtually every programming language supports numbers, strings, groups of things (objects/records/structs), and recursive composition of information. These same fundamental structures appear repeatedly because they reflect the way humans organize information. These same structures appear in every GUI tool kit for the same reasons. Although the dialog structure and supporting information required for spoken-language information will be quite different in a USI, those same fundamental information structures will appear. A USI built around those structures should be effective in achieving the same transference of skills and generality of application as a GUI.

Research Agenda

The universal speech interface approach to spoken-language interfaces is a departure from the natural language-based approaches taken by others. The USI approach assumes that, at least for simple machines, the advantage of speech lies not so much in its "naturalness" but in interactive contexts where screens, buttons, and other, more traditional interaction forms do not apply. A hypothesis contrary to ours states that natural language-based speech will be more effective than a uniform style. Another contrary hypothesis is that some other interactive modality such as buttons, lights, projectors, or cameras will be more effective in achieving ubiquity of interaction. The testing of these hypotheses is the heart of an interesting research agenda.

In comparing natural language with a constrained language the key measures are learnability, transfer of learning, and robustness and expense of the technology. The key contention for natural language is that transfer of speech expertise from the human-human to the human-machine world will mitigate most learning and transfer problems. The contention for a USI is that the inherent limitations of each simple machine will produce uncertainty and confusion among users of a new natural-language interface. Early evidence supports the assertion that users prefer a less efficient but more predictable and trusted interface [11]. It is hoped that standardizing the interface will give users an efficient path for discovering and using simple machines by exploiting transfer among human-machine interfaces rather than from the human-human domain.

Initial learnability is also a key issue. Natural language presumably requires no initial learning. Any deviation from natural language will require initial training. First-time GUI users do not understand double-click, right-click, or drag, but once they do, they have tools. A USI must have a shallow learning curve to be successful. A good example of this approach can be found in the Graffiti™ character-writing system currently deployed with the PalmPilot PDA. Graffiti is an invented "language" that was engineered to optimize automatic handwriting recognition. It strongly resembles the way people naturally write letters, digits, and punctuation marks. The original, natural style of writing was modified minimally, only when it was prudent to do so for ease of recognition. As a result, it takes only a few minutes to become a productive Graffiti writer. Similar approaches may work in speech.

The only way to resolve the debate over natural versus constrained spoken language is by constructing many prototypes and a great deal of user experimentation. Researchers in user interface management systems (UIMS) worked for years on a wide variety of approaches to inexpensive development. Most of them were discarded before widget libraries and interface development environments (IDE) such as Visual BASIC became wildly successful. There is no reason to believe that the speech tools research process will be different. A key measure for the technological success of a USI will be the breadth of applications that its restricted language can effectively service. Many GUIs are very easy in Visual BASIC and yet there are common types of interfaces that are very difficult within its framework. A test of the USI tools will be whether they can develop a large enough class of applications, cheaply enough to diminish the impact of those limitations.

A good USI system should have:

- ✶ A shallow learning curve. This will help users to get started with readily accessible mechanisms to build expertise both in the application and in the USI concepts themselves. GUI users do not learn command-key shortcuts at first, but when they do learn them, their effectiveness in using all applications increases.
- ✶ Standardized ways to accomplish standardized tasks. This is the key to the transference.
- ✶ Reliable and predictable handling of the vagaries of speech recognizers.

Serious experimental questions must also be answered when comparing speech and other means for interacting with simple machines. The ambient nature of speech gives it both advantages and disadvantages over visual and manual techniques. The nature of when and how these trade-offs occur is not well understood. The work of Oviatt [6] sheds some light on these questions. Speech has a much wider breadth of expression, but buttons are more reliable and responsive. Most of these comparison questions remain unanswered because research in speech application tools is lacking. The advent of inexpensive recognition technologies opens this fruitful area of research.

Summary

By its nature, speech opens up interactive possibilities that surpass graphical user interfaces. We believe that understanding how the possibilities of spoken-language interfaces are separate from the natural language problem will produce many more spoken-language interfaces than ever before. In many situations interaction by text, pointing, or using a display is not feasible or desirable. In such cases, speech is indispensable. When graphic interaction is possible, speech can still take its place alongside it. Research into the nature of speech interfaces and their tools is entering an exciting new phase. We have articulated here one vision of what speech interaction between humans and semi-intelligent machines could be like in the foreseeable future. Information on our ongoing efforts in this area can be found at www.cs.cmu.edu/~usi and www.cs.byu.edu/ice. However, these can represent only a handful of points in this vast design space. It is our hope to see many more attempts in this direction.

References

1. Arons, B. Hyperspeech: navigating in speech-only hypermedia. Proceedings of the Third Annual ACM Conference on Hypertext (1991), pp. 133–146.

2. Jacob, R. Using formal specifications in the design of human–computer interface. *Proceedings of Human Factors in Computer Systems* (March 1982), pp. 315–322.

3. Kamm, C., Walker, M., and Rabiner, L. The role of speech processing in human computer intelligent communication. *Speech Communication 23* (1997), pp. 263–278.

4. Newman, W. A System for Interactive Graphical Programming. SJCC. Thompson Books, Washington, D.C., 1968, pp. 47–54.

5. Olsen, D.R. Interacting in chaos. *Interactions* (Sept. 1999), pp. 42–54.

6. Oviatt, S. Ten myths of multimodal interaction. *Communications of the ACM 42*, 11 (Nov. 1999), pp. 74–81.

7. Price, P. Evaluation of spoken language systems: the ATIS domain. *Proceedings of the Third DARPA Speech and Natural Language Workshop* (R. Stern, ed.), Morgan Kaufmann, June 1990.

8. Rosenfeld, R., Olsen, D., and Rudnicky, A. A Universal Human–Machine Speech Interface. Technical Report CMU-CS-00-114, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, March 2000.

9. Rosenfeld, R., Zhu, X., et al. Towards a universal speech interface. In *Proceedings of the International Conference on Spoken Language Processing,*, Beijing, China, 2000.

10. Shriver, S., Black, A.W., and Rosenfeld, R. Audio signals in speech interfaces. In *Proceedings of the International Conference on Spoken Language Processing*, 2000.

11. Walker, M.A., Fromer, J., et al. What can I say?: Evaluating a spoken language interface to e-mail. *Human Factors in Computing Systems* (April 1998), pp. 582–589.

12. Zoltan-Ford, E. How to get people to say and type what computers can understand." I*nternational Journal of Man-Machine Studies 34* (1991), pp. 527–547.

13. Zue, V. From interface to content: Translingual access and delivery of on-line information. *Proceedings of Eurospeech'97*, Rhodes, Greece, 1997, pp. 2047–2050.