

While [interactive speech] technology improves constantly, it is unlikely that, in the foreseeable future, it will approach the robustness of computers in science fiction movies.

— Jennifer Lai and Nicole Yankelovich

You've seen it in Star Trek, Mission Impossible and James Bond — futuristic technologies based on voice recognition and verification. Long the stuff of science fiction fantasy, voice technologies are now a business reality thanks to Nuance.

— Nuance Communications (www.nuance.com, 2002)

Toto . . . I have a feeling we're not in Kansas anymore.

— Dorothy

Introduction

The user interface to an interactive product such as software can be defined as the languages through which the user and the product communicate with each other.

— Deborah Mayhew

Interfaces

User interfaces (hereafter, mostly just “interfaces” with various specifying adjectives, like “voice” and “graphic”) are the media by which people interact with digital systems. Although it is an abstract and quite shallow image (in particular, an interface is part of the system, not outside it, and strictly speaking, so is the user), it is nevertheless useful to picture the interface in the terms of Figure 1.1: A user performs some physical actions according to an established protocol, effecting input, which trigger electronic operations. Almost always, these operations include indications for the user about the effect of her actions; that is, feedback.

Voice interfaces are interactive media in which the input is primarily or exclusively speech, and so is the feedback. They are new phenomena, especially in the conversational style that dominates the approach of this book, and their design draws on wide bodies of language research, from linguistics, philosophy, sociology, and psychology. The results of this research, in turn, need to be interpreted, then applied, through the disciplines of computer science, telephony, and interaction design.

Voice interfaces are so new, indeed, that their properties and possibilities are still being worked out, daily. So new, that there is still only a partial awareness among designers that those properties and possibilities need to be investigated in a way that is independent of specific systems and technologies. So new, that — even though there is widespread

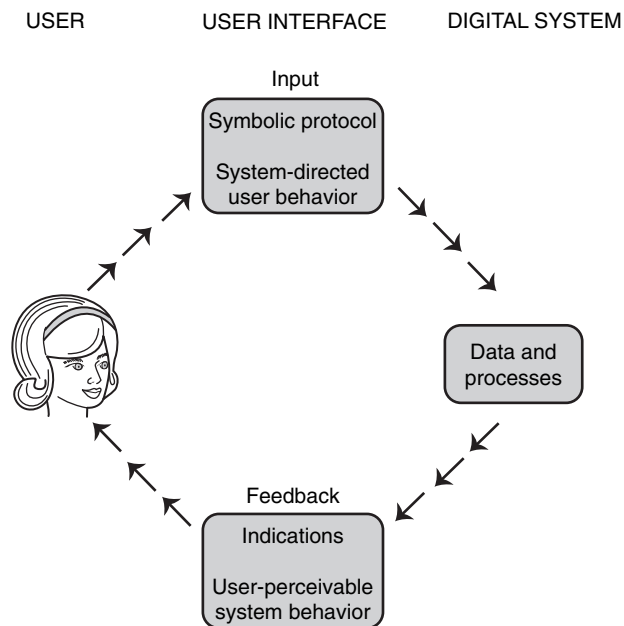


FIGURE 1.1 A rough schematic of user interfaces

recognition that building usable speech systems overwhelmingly implicates the field of human–computer interaction — the word *interface* is not often used by the very people who are developing them. “Spoken Language Systems” is more common, along with a variety of terms featuring “dialogue” — like “Automatic Telephone Dialogues” and “Spoken Dialogue Systems.” These labels reflect much the same attitude that characterized the early years of graphic interface development, which was subsumed under general application development and often designed by the same software engineers who designed the system mechanics.

Voice interfaces are among the growing range of interfaces populating the modern landscape, from the rapid, virtually invisible interfaces of digital calculators to the awkward on-screen interfaces of digital televisions, most of them exploiting multiple modes (chiefly, sound, vision, and touch). But three specific interface categories are significant for the evolution and design of voice interfaces. Two are familiar from computer interaction: the nearly archaic command-line interface and the ubiquitous graphic interface. The third comes from telephony: the keypad interface, which provides interaction with messaging applications, automated reception systems, telephone banking, and the like. It is in these domains, where keypad interfaces have dominated for a decade or more, that voice interfaces are beginning steadily to appear. An inevitable and near-total eclipse of keypad interfaces is around the

next bend (how far ahead that bend is depends on economic and technological developments; but it *is* the next bend).

Since all three of these interface types are implicated in voice-interaction design, and since I will be drawing analogies from them throughout the book, we'll look at them briefly in turn, and also glance at the related area of multimodal interfaces, where voice may come to play an increasingly important role.

Command-line Interfaces

149. Implementation restriction: The “stringrange” on-condition cannot be enabled when the “substr” pseudo-variable is used. Note that this restriction does not apply to the “substr” built-in function.

— A PL/I Multics error diagnostic

Command-line interfaces were usually just called “Man-Machine Interfaces” (MMIs) in their period of dominance, the 1970s. The name reflected both the gender imbalance that characterized computer development and the fact that they were the only game in town. As in Figure 1.2, they work on a linguistic paradigm (strings of alphanumeric “words” arranged in a determinate syntax).

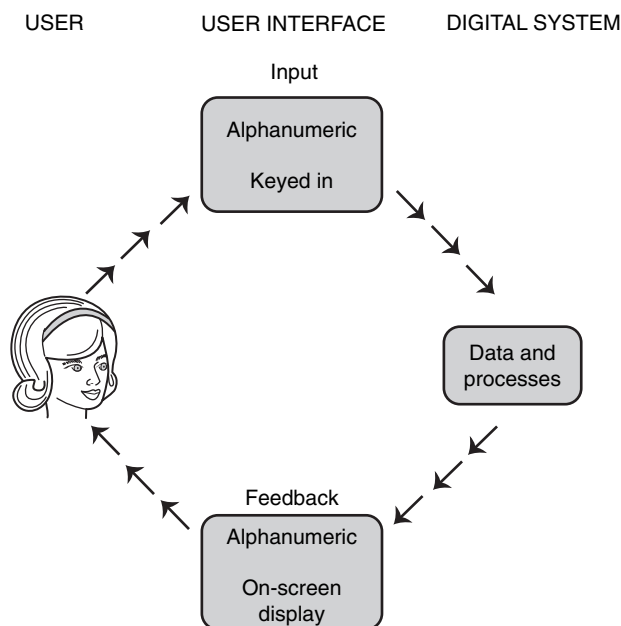


FIGURE 1.2 A rough schematic of command-line user interfaces

Users issue typed commands and receive textual feedback, resulting in interactions of the following sort:

```
User  $MESSAGESYSTEM RETRIEVE
MTS   Mailbox EYVQ: 1 new, 4 old messages
```

That particular specimen is the command users once issued to MTS (the Michigan Terminal System) to get a list of email, followed by MTS's response.

Command-line interfaces like MTS or the better-known Multics (Multiplexed Information and Computing Service) could be tremendously efficient, compressing a whole range of operations into one sleek line of text, so long as the user knew their narrow vocabularies and rigid syntaxes. But these interfaces incorporated very little sense of the user, and were brutally unforgiving. In particular, there was comparatively negligible attention to the clarity of the feedback. The system either did what you told it to, with minimal confirmation of its actions, or it generated an error message, often cryptic in the extreme, to account for why it wasn't doing what you thought you told it to do. Here is a from-the-trenches characterization of the user's relation to the system, during the heyday of command-line systems:

The user is often placed in the position of an absolute master over an awesomely powerful slave, who speaks a strange and painfully awkward tongue, whose obedience is immediate and complete but woefully thoughtless, without regard to the potential destruction of its master's things, rigid to the point of being psychotic, lacking sense, memory, compassion, and — worst of all — obvious consistency.

(Miller and Thomas, 1977: 172)

They weren't completely lacking in goodwill, however, and some subsystems even showed a measure of personality. I was once fortunate enough to get, upon issuing the following command, the next-following response:

```
Me    $MESSAGESYSTEM RETREIVE NEW
MTS   Didn't your momma ever tell you: I before E, except after C?
```

The last bastions of this era are MS DOS (Microsoft Disk Operating System), which has been overlain by various incarnations and generations of Windows, and Unix (named by way of an arcane, nose-thumbing pun on *Multics*), which undergirds (among other graphic interfaces) Linux and Mac OS X. While command-line interfaces still have active user populations in some specialized communities, and while the graphic overlays sometimes have to be peeled back to the command-line level when things go wrong, or when higher efficiencies are needed, they are powerful yet lumbering dinosaurs; graphic interfaces have clearly inherited the earth.

Graphic Interfaces

Those buttons, graphics, and words on the screen through which we control information.

— The explication of “interface” in the jacket blurb to Steven Johnson’s
Interface Culture

Graphic interfaces are often just called “interfaces” these days, so much have they become the default, but also GUIs (Graphical User Interfaces). They work on an object-manipulation paradigm (representations are moved, altered, or engaged — sometimes all three — through the user’s wielding of a pointing device, usually augmented by physical buttons or keys) as outlined in Figure 1.3.

Users interact with systems, usually in clusters, represented by dedicated areas of the screen looking something like the object in Figure 1.4. That particular specimen is from the earliest web browser, Mosaic. It has the now-familiar doodads: buttons along the top for back, forward, home-page, refresh, and save; along with a scroll bar; links; a text field for keyboard input; and a primitive menu (the box at the top with the down-facing arrow head provides for navigation among previously visited sites). The display is graphic, the

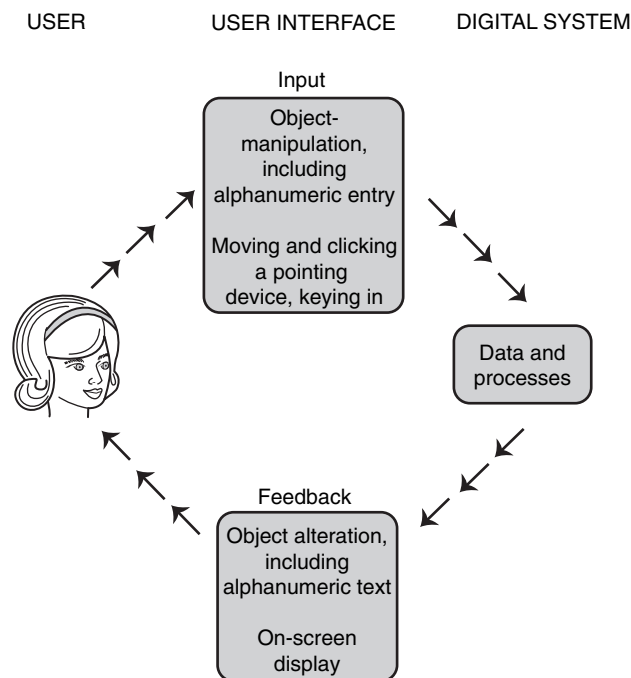


FIGURE 1.3 A rough schematic of graphic user interfaces

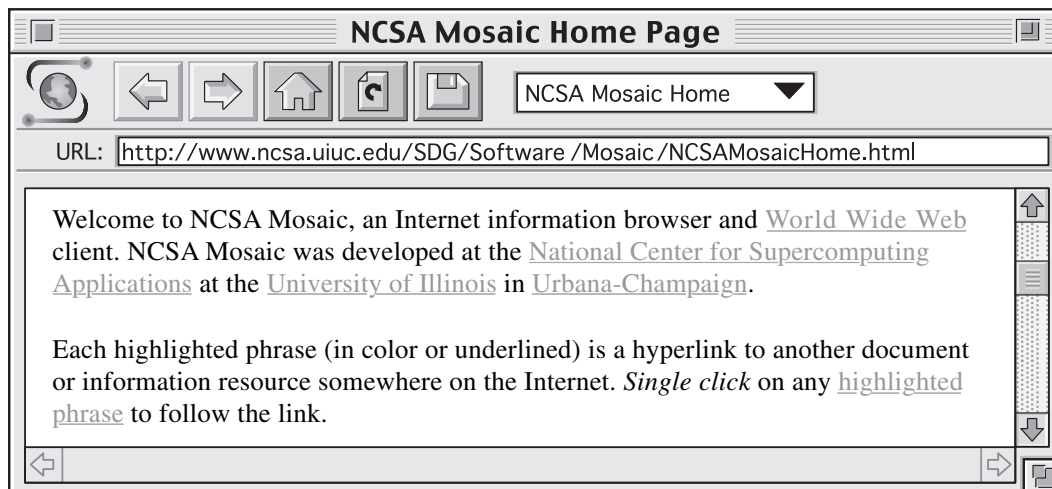


FIGURE 1.4 A typical early graphic-interface window

input is direct manipulation. When graphic interfaces were still widely viewed as second rate in the (much smaller and snobbier) computer community, they were awarded the acronym WIMP, which, ignoring its derisive connotations, still captures the interaction style fairly well: Window-Icon-Mouse-Pointer. (Now that other input/output modalities are under widespread exploration, the acronym has come back to life; the most familiar adjective for alternative interfaces is “non-WIMP.”)

Command-line interfaces have not, of course, disappeared under the dominance of WIMPs. As I mentioned, a few have survived in peripheral communities (Unix is the most notable of these). But, more significantly, command-line interfaces can be seen in broken and spectral form haunting some functionalities of graphic interfaces. One of the most familiar elements of graphic interfaces is the menu, which includes the (naturalized, rationalized, and contextualized) names of commands; buttons, too, are commands; field labels cue parameter entry; and so on. Graphic interfaces have added imagery and a more expansive sense of space, separating and distributing the functions of the command-line interface, but bits and pieces of the latter live on in every cranny and nook of the former.

Graphic interfaces are about to undergo a similar fate.

Multimodal Interfaces

During multimodal communication, we speak, shift eye gaze, gesture, and move in a powerful flow of communication that bears little resemblance to the discrete keyboard and mouse clicks entered sequentially with a graphical user interface.

— Sharon Oviatt and Philip Cohen (2000)

The era of the graphic interfaces is nearing an end, but they too will live on, as part of multimodal systems with increased physical inputs and a wider range of outputs — in both cases, chiefly motion and sound. On the input side, gesture and speech will become increasingly important; on the feedback side, video and sound (prominently including speech).

While multimodal interfaces will soon be ubiquitous, and while the most important incorporated modality will certainly be voice, I don't explore the combination of voice with other input/feedback modes in this book, except in passing. Many of the principles of pure voice interfaces we take up are of course applicable to speech in multimodal design, especially the discussion of language that occupies Part I. But I mention multimodal interfaces here largely to explain their absence from my explicit concerns in this book.

I concentrate on voice-only interfaces, with a few casual remarks about voice as a modality in multimodal systems, for two reasons. Designing speech interaction as one modality among several is too easy. And it is too hard. Speech as a modality is too easy because the redundancy of context puts far less pressure on speech design. As one simple example, response delays in speech when that's the only channel are confusing and highly annoying. Think of listening on a telephone to seconds and seconds of silence after you've asked a question. You ask again, but still there's no response. Being put on hold to the caramel tones of a further-homogenized Bee Gees tune only makes it worse. But if you ask a question of someone in person, and they hold up their hand to indicate you should please hang on for a moment — while they catch your eye to suggest they just have to finish up with this other task first — you're far more liable to be patient. I am.

Multimodal systems have this characteristic. They don't require you to focus all your attention on one channel for feedback, a channel that is thereby blocked to further activity while you can only sit and grind your teeth. A graphic display can show that processing is going on, or that an external data source is causing a logjam — some explanation and some temporal indication as to when you can return to the activity — while the multitasking capabilities inherent in such systems let you do something else in the meantime. The other modalities augment speech.

If you can design a voice-only interface, designing the voice elements of a multimodal system is, if not exactly a piece of cake, certainly much easier than the reverse: moving from voice-modality design to voice-only design. Similarly, concerns such as conversational style, continuous recognition, and speaker-independence are all much less urgent for multimodal systems, since there are enough other channels available that individual words or short phrases can do efficient work for the user without her concerning herself much with nuance.

But designing speech as one modality among several is also too hard, in another way — because of the incredible context-sensitivity of language. In a multimodal, multitasking environment, in an office where there are phones ringing, other people talking, and your attention may be widely distributed, something as (seemingly) simple as the referent for a

pronoun, or as (seemingly) determinate as the topic of a discourse, can be extremely difficult for humans to peg down, never mind for machines.

On that note, let's return to our rapid survey of interface styles.

Keypad Interfaces

- 81# *Menu of the Touchtone Teller*
- 51# *Current Certificate and Savings Rates*
- 52# *Current Loan Rates*
- 11# *Savings Account Balance (01)*
- 12# *Checking Account Balance (02)*
- 15# *Specific Account Balance (Plus 2 digit Account Number)*
- 16# *Loan Balance (Plus 3 digit Loan Number)*
- 9# *Last ATM or Electronic Withdrawal or Transfer*
- 19# *Interest Paid on Loan*
- 20# *To Access Another Member Account*
- 66# *Change Access Code*
- 69# *Card Activation*
- 80# *Repeat Last Response*
- *# *Cancel Transaction*
- 99# *End Call*

Some codes from Touchtone Teller Star One Credit Union

Keypad interfaces, shown in Figure 1.5 and sometimes called TUIs (Touchtone User Interfaces), work on a linguistic paradigm, employing “words,” usually in isolation, though sometimes in a determinate syntax. They consist of recorded instructions or queries that users respond to with (literal) button pushes.

A typical exchange looks like this:

Phone: To make your withdrawal from your savings account, press “1.” From drafts, press “2.” From shares, press “3.” From a loan advance, press “4.”

User: beep

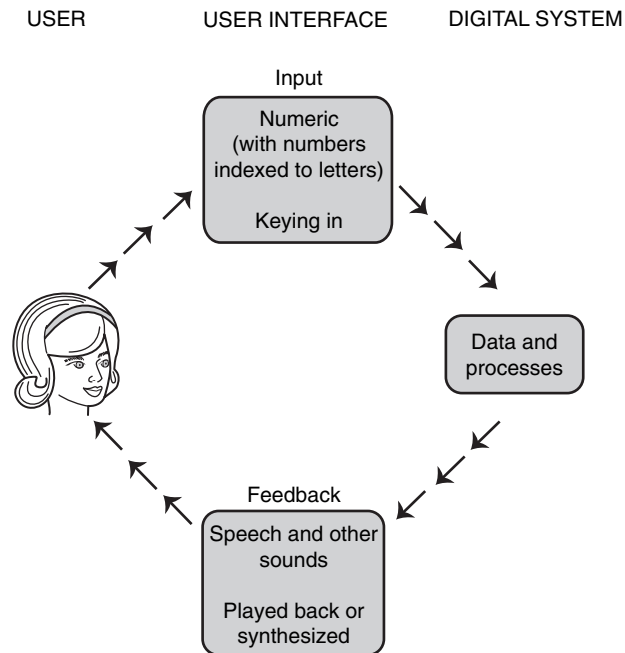


FIGURE 1.5 A rough schematic of keypad user interfaces

As hard as the designers work, as ingenious as some of their solutions are, and as diligent as their user research is, these systems are the bane of most telephone users' existence, and their poor satisfaction rating is one of the principal motivations behind the development of voice interfaces. The catalogue of their annoyances include:

- Arbitrariness of key-function relationships: why does 1 mean “savings,” 2 “drafts,” 3 “shares,” 4 “loan advance”?
- Transience of the key-function relationships: with the exception of a few stable key-function match-ups (* might always mean “main menu” in a given system; 0 might mean “get me a real human”), for each successive exchange in the encounter the keys have different functions (1 means “banking,” then it means “withdrawals,” then it means “savings,” . . .).
- The size, number, and arrangement of the keys: there are too few (only twelve active commands can be available at any given time), too densely packed, in a small matrix rationalized only for short, infrequent, and unidirectional numeric input, not for even brief interactions.

- The location of the keys: most keys are on the handset nowadays, which has to be held up to the head for feedback, then away from the head for input, then up to the head again for feedback.
- The seemingly interminable, hierarchical navigation trees (“menus”): after the user presses 1 (or 2 or 3), there is another branch, and another; then the user might be forced to climb back up the tree and take another branch — a gagged and bound (but for one finger) monkey, trying to follow the mindless, one-note commands of a ruthless task-master.

Speech combats all of these annoyances by its very nature. Words have meaning. That’s their job. Meaning doesn’t have to be assigned. (Alas, however, early voice systems did indeed have prompts like “for transportation, say ‘one,’” sadistically avoiding the natural semantics of words.) And there are a lot of words to choose from, not just twelve. And the user’s head is deployed for both the production of input and the reception of feedback. And language is largely independent of conceptual structure; it can instantiate hierarchies, or level them.

The specific solutions to most of these problems, however, are not automatic with the use of language. Aside from the fortuitous production/reception properties of the human head, overcoming these problems falls to implementation and design. While languages have huge numbers of words, systems don’t. Their vocabulary space is limited, and one of the most critical jobs of voice interaction design is allocating that space efficiently. Also, while hierarchies can be apparently flattened, they remain very important architectural strategies in system design. Voice interaction design often involves not an organizational structure which avoids hierarchies, but an interaction style which doesn’t overly determine the users’ paths through them.

Voice Interfaces

No longer the sovereign property of humans, speech has become an ability we share with machines.

— Sarah Borruso

Voice interfaces have a range of labels — as above, many of them are configurations of “spoken,” “dialogue,” and “system,” with each other or related words, usually reduced to acronyms (see the glossary for the most common terms). *VUI* (Voice User Interface) has recently emerged as the leading short-hand term. They work on a linguistic paradigm (word strings), and consist of utterances, plain and simple: speech in and speech out, as outlined in Figure 1.6. Niels Ole Bernsen and Laila Dybkjær define a pure voice interface as “a *uni-modal speech input/speech output* system which conducts a dialogue about a *single task* in a *single language* with a *single user* at a time” (2000). (Their preferred term, actually, is SLDS — for Spoken Language Dialogue System — but it overlaps so closely with what I am calling a “voice interface” that I am comfortable poaching their definition.)

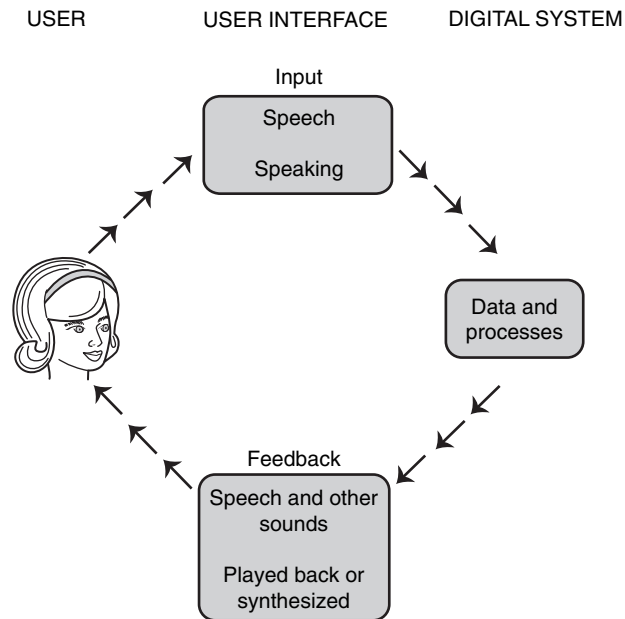


FIGURE 1.6 A rough schematic of voice user interfaces

While we humans are pretty good at working in multiple modes, and many of us are good at multitasking as well, the voice interface's unimodality is one of its chief sources of power. The other is the fact that the relevant modality, speech, is one to which humans are very highly adapted.

Speech in, speech out: the caller says something, the system says something back. Here's an example:

SpeechActs Welcome to SpeechActs. Please say your full name.

Nicole Nicole Yankelovich.

SpeechActs You currently have "Lunch with Kate Ehrlich" until 2 pm. At 3 pm you have "Brainstorming meeting in the lab." At 6:30 pm you have "Dinner with Ellen."

Nicole What do I have tomorrow?

SpeechActs Tomorrow, you have "Dentist appointment."

Nicole What about Bob?

SpeechActs On Tuesday, November 16th, Bob has “Visit to Georgia Tech.”
 Nicole And on Wednesday?
 SpeechActs On Wednesday, November 17th at 2:15 pm, Bob has “Return to Boston.” At 5:15 pm, Bob has . . .”
 Nicole Stop. Tell me what I have.
 SpeechActs On Wednesday, November 17th you have no appointments.

(Yankelovich, 1994)

This specimen comes from a first-generation conversational voice interface developed at Sun Microsystems in the early 1990s to provide telephone access to basic computer tasks such as e-mail and calendars. (In addition to Yankelovich, 1994, see also Yankelovich, 1995, 1996, and Martin et al., 1996.) While this was an in-house system and more of a test-bed than a product, and consequently primitive in many respects, you can probably see some of its virtues at a glance:

- The caller has much more initiative in the flow of the interaction. Nicole is in control.
- The application allows for implicit reference. Nicole does not have to say “What about Bob’s schedule tomorrow?” A simple “What about Bob?” does it.
- The logic of the application is consistent but not bullheaded. Close to the end of this sequence, for instance, SpeechActs starts telling Nicole about Bob’s Wednesday appointments, which is reasonable, given that the topic of the exchange at this point (raised by Nicole in her previous turn) is Bob’s appointments. Once Nicole realizes she and SpeechActs aren’t on the same page, though, she can repair the problem quickly.
- The hierarchical structure is opaque (though architecturally still there): Nicole isn’t forced to navigate the structure stepwise. She really wants to know about her own appointments, for instance, when SpeechActs starts telling her about Bob’s, and the application lets her get elegantly to her own calendar. She gets there not by explicitly going back up to a “calendar” menu fork, and then down the “Nicole Yankelovich” branch. She just says “Stop” and SpeechActs drops this line. In the same turn she adds “Tell me what I have” and the system — retaining both the topic “calendar” and the focus on Wednesday, November 17th — responds with information about her schedule.

And, most importantly, the feature that the others all hang on:

- The language the caller can deploy is far more natural. It’s command-like in places, more abrupt than most of us would probably talk to another person (“Stop. Tell me

what I have.”). But *SpeechActs* is not a person, and does not take offense. It’s not the tone (“brusque” here, rather than “polite”), however, that makes the interaction linguistically natural. It’s the ability to collapse a sequence of system actions (or states) into one ordinary-language utterance (not a specialized syntax of isolated operators), which gets its meaning from context as much as from dictionary entries for the words. And, of course, there are also utterances that would be perfectly at home in a human-human dialogue (“What about Bob?”).

Conversational voice interfaces are not the only type of voice interface; they are the preferred voice interface for most phone-based interactions, and they are the focus of this book, but they are computationally expensive and creatively demanding, and are barely known among the general public. If you ask the general public about a “speech application” they might have encountered, they are likely to think of menu-driven voice systems that share the structure of keypad interfaces, and they are likely to curse.

These far-more-common speech applications behave like this:

Telefónica For information regarding transportation, say “transportation.” For information regarding entertainment, say “entertainment.” For information regarding weather, say “weather.”

Caller Transportation.

Telefónica For information regarding air travel, say “airport.” For information regarding bus travel, say “buses.” For information regarding rail travel, say “trains.”

Caller Airport.

Telefónica For arrivals, say “arrivals.” For departures, say “departures.”

And so on. And so on. And so on.¹

These applications — voice-response systems — have their virtues, and involve many design subtleties. Done right, they can eliminate the arbitrariness and transience of function-semantics that plague keypad systems. They have available vocabularies substantially bigger than 12 keys. And they efficiently capitalize on the human head’s dual input-output capacities. For these reasons, voice-response interfaces are an important advance over

1: This dialogue is extrapolated from an early application of AT&T’s Intelligent Network service deployed in Spain (called, as in the text, *Telefónica*), reported in Jacob et al. (1992) and Wilpon (1994). The extrapolation takes some poetic license, but to the advantage of the original system, which was even more annoyingly tied to the keypad design legacy. The actual prompts in fact directed the caller to say “uno” for transportation, “dos” for entertainment, and so on.

keypad interfaces, but their call-and-response, one-function-at-a-time interaction, and the hierarchical structure that determines the interactive sequence, makes them tedious to use.

Voice-response systems are not evil, despite the rants I have heard from users, and occasionally given vent to myself. Even the dreaded keypad systems are not wholly without merit (if you know them well, or have a good cheat sheet, they can be very efficient). Indeed, designing conversational voice interfaces needs to build upon voice-response interactions, not start from the ground up, and sometimes conversational interactions will proceed indistinguishably from voice-response interactions. Sometimes conversational breakdowns require voice-response remedies. Sometimes, even the keypad should be integrated into a conversational voice interface (for security, in particular, or robustness when signal quality is low).

But the advantages of conversational interaction are so substantial, because the available resources of human–human vocal communication are so rich, that it is far-and-away the best design style. And it is the topic of this book.

Why Speech?

Admittedly, the keyboard option might be less practical at this moment. . . .

—Caption on a 2002 ad for Vbox (a small, hand-held, voice-and-keypad input device), on a small poster mounted above a urinal.

Implementing voice interfaces to eliminate the scourge of the keypad and counteract the curse of the hierarchy has far-reaching implications, and not just for our collective sanity as a user population. The increasingly competitive call-center sector, for instance, is a commercial area with an economic meter ticking in the billions every year. Voice systems will inevitably become more and more important in this sector, as they already have (along with keypad systems) in phone platforms of most other commercial operations, and ease of use will quickly separate the successful systems from next week’s boat anchors. Conversational design is the (potential) road to usability.

Why speech? Largely because the alternative modalities for portable information devices lead to interactive abominations. Telephones are ubiquitous, information is abundant, and there are many times when the former is the best (most convenient, most portable, safest) way to access the latter. But even souped up with liquid-crystal displays, wheels, soft keys, memory cards, features, features, and more features — whatever their truly ingenious designers can glue on — there is an input/output bottleneck that hardware can never solve. The screens, if they are small enough to be genuinely portable, are too small for much information output, and the keys are too small for our gross digits to punch in much information. As *New York Times* technology reporter, Katie Hafner, laconically put it, “browsing the Web with a five-line screen and a tiny number pad is not a very gratifying experience” (Hafner 2002).

The Japanese word for the hordes of mobile informavores roaming our landscapes is *oyayubizoku*, which translates as “the tribe of the thumb,” for the digit they are always poking at their portable devices. Thumbs are good. Their opposability has served us hominids very well. But as organs of communication, they fall vastly short of mouths. Given the choice, most people would go with voice over appendages for input into itsy-bitsy devices, any day.

As for output, the screens, even at their small dimensions, certainly have potential. We humans are very efficient visual processors. Information fed to us along that modality can be very rich. But there are still considerable difficulties. Not all information is best in visual format. I would rather hear “warm and hazy with a thirty percent chance of afternoon thundershowers” than try to puzzle out a series of pictograms, especially if I had to squint at a miniature screen to make them out. And even information that might be best presented visually, I would still prefer to hear if it meant I could keep my eyes on something more urgent, like an eighteen-wheeler coming up fast in my rear view mirror.

There are also significant barriers to graphic interfaces that voice interfaces can overcome, both individual and situational. Not everybody is equally facile with graphic interfaces. Some people can’t see, or can’t see very well. Some people can’t point and click and type (a category which includes not just the severely disabled but the increasing legions of computer users with repetitive strain injuries). Some people, often called “children,” don’t have the dexterity, or even the hand size for keyboards and pointers. And not every environment is equally amenable to direct-manipulation input. Computer systems would be better served by voice input and output when they are used in situations where hands and eyes might otherwise be better occupied — repairing equipment, for instance, or sorting through inventory, or just (in the case of hands) jammed into pockets or mittens for warehousing operations in a North Dakota January.

Speech is, further, very resilient as a side channel, making it the ideal mode for what Salvucci (2001) calls *secondary-task interfaces*. These are interfaces for systems or functions when the computational activity is not the primary task (for instance, supporting an installation where the user is busy handling equipment but still needs to check part numbers, follow procedures, and the like). The issue of safety can become quite important for secondary-task interfaces when the primary activity is potentially dangerous, such as driving. As long as the guy screaming past me on the highway and switching lanes with abandon insists on using his telephone, I would be happier if he had voice dialing. As long as he wants traffic or weather or stock information on demand, while he is switching lanes, I would rather have him talking and listening than trying to push buttons or squinting at a tiny display.

Speech also has the potential to reduce the mode confusion common in everyday digital living. One of the most compelling attractions of *Enterprise*-style life-in-the-future is the ability to tell the elevator what deck you want to go to, to tell the holodeck to play sultry music or end program, to tell your replicator “Tea. Earl Gray. Hot.” Sure, the holodeck and

the replicator would be handy, but it is homogenization of the interfaces — just talk — which makes that world so seamless. As most speech-system researchers will tell you, shortly after they say hello, the world of *Star Trek* is a long way off (it does take place in the 24th century after all). But the possibility of interacting with, say, entertainment appliances “without first scanning a cryptic array of choices on any of several remote control devices” (Hafner, 2002b) is imminently possible.

Speech also opens up more than just input and output channels. It can open up an additional cognitive dimension. Affective computing is becoming a significant research pursuit, and voice is a more reliable indicator of emotion than anything available to a graphic interface. Word choice can signal mood quite directly (“great” suggesting more enthusiasm than “good,” for instance). But the pace and style are even more reliable. Rapid volume changes suggest anger or urgency; slower, more deliberate speech correlates with seriousness, chirpy speech with pleasure. To the extent that computers will make headway either interpreting or inducing emotion, they will need a fuller understanding of speech.

Why Conversation?

The brain is actually a very shitty computer.

— Richard Wallace

We’re good at conversation, we humans. True, some people are more entertaining at dinner parties than others, and some people know how to make you feel more interesting when you talk to them than other people do. Some people you enjoy conversing with, some you don’t. But all of them are “good at conversation” in the same fundamental way of knowing that a greeting pairs with a greeting, but a question pairs with an answer; that drawn-out intonation means the speaker isn’t ready to relinquish his turn to talk just yet, even though he may have run out of words; that specific routine strategies of coherence and cohesion weave notions into topics, words into discourse. There is variability in our capacities to deploy this conversational know-how, just as there is variability in the velocity and grace with which we move, but we are all masters of conversation in very much the sense that we are all masters of bipedal locomotion. It’s something we do, after early childhood acquisition, instinctively.

Computers, thankfully, are fairly good at actuating our conversational dispositions, in limited, task-specific ways. It is not part of their essential structure, which really only carries out computational instructions — changing states, shuttling information around, setting conditions, and the like. The facility of computers to work with our conversational intuitions is the result of decades of difficult, user-centered interface crafting; of developing systems that accommodate our native human strategies and talents rather than (just) requiring humans to accommodate the single-minded, instruction-processing dispositions native to computers. At the very outset of real-time, individual-user, responsive comput-

ing, programmers quickly adopted a conversational metaphor for this new mode of computer behavior (Orr 1968, vi). Interactive design begins with this metaphor. One of the foundational articles in the evolution of graphic interfaces (Foley and Wallace, 1974) charted the significant linguistic analogs in graphic interfaces, stressed the critical importance of feedback, and argued for a design philosophy anchored in the concept of “Natural Graphic Man–Machine Conversation.” The importance of this philosophy for the success of graphic interfaces is hard to overestimate; it governs the systematic tendency to treat most user actions not just as instructions to be processed but as contributions toward a goal, contributions requiring a semantic response.²

But it is a metaphor. We pass the pointer over a tool and get the semantic pop-up response “Text Box” or “Insert Picture.” We key command+S and get the semantic response of a highlighted File menu and little jingly sound. On the other side of the interaction, our machines await instructions disguised as semantic input like pressed “Open” or “Print” action buttons. The computer doesn’t care about those labels, only about the microcode they get translated into. For the machine’s purposes, they could be reversed, or “Geöffnet” and “Druck,” or “Archie” and “Jughead.” The semantics are for the humans.

Our dealings with computers are rife with pieces of natural language, not just in button labels and menu-item names, but in “dialogs,” often defined by the conversational turn structure of standard utterance-pairs like question and answer. “Do you want to save the changes you made?” my word processing application asks me when I quit, soliciting any of three responses, “Save,” “Don’t save,” and “Cancel” (and biasing for “Save”). This sort of interaction would never be mistaken for a chat; the metaphor only goes so far, and most of the time you wouldn’t want it to go any further. Graphic interaction is *similar* to conversation in important ways, but it is not a *type* of conversation, and it shouldn’t be.

Now, however, we have computers that can process speech and respond in kind, whose input-output semantics therefore come almost exclusively from natural language (rather than partially and metaphorically), and whose ear-and-mouth (rather than hand-and-eye) interactivity triggers our conversational dispositions more immediately and insistently than any other technology we currently have. The question is, then, should we thwart these dispositions or court them?

The options in the design of these new interactive creatures are subtle and varied, and we will explore them in some depth, but the question of an overall design philosophy can be oriented with respect to two poles: the keypad model, where the machine presents options, constrained by the extremely limited input options to resemble a multiple-choice exam; and the conversational model, where the interaction occurs between agents working towards a mutual goal.

2: By *semantic response* here, I mean only that the system responds not just with an internal change of states but with a meaningful signal to the user about the change of states. In particular, I do not mean “semantic” in the Foley and Wallace sense of specifications of system functions.

Should these voice systems, that is, lead us about by the tongue, utterance by utterance, up and down the ladders of an obscure hierarchical network? Should they demand that we repeat isolated little noises they dictate to us, a staggeringly atypical pattern of talk? Should they respond to us with repetitions of their own, parroting our words back to us incessantly for confirmation, like a four-year-old kid trying to drive her brother crazy? Or should they function negotiatively — making suggestions, but also following our lead, initiating but also responding to communication repairs, interacting in humane and sensible ways? Interaction design is a tradeoff, always, between machine and human. But the keypad model favors the machine in that tradeoff; the conversational model favors the human.

The interactions that result from good implementations of the conversational model lose their metaphorical quality. They are not *like* conversations, they *are* conversations. They are not freewheeling dinner party conversations. They are more on the order of service-encounter conversations, where one person helps another with some information or assists their purchase of a hammer.

“Before too long,” Jef Raskin quotes a mobile computing authority, “you may not have to worry about an interface at all. You may find yourself simply speaking to your computer” (1993: 122, 2000: 2). Raskin reproves this authority for failing to understand that voice interaction *is* an interface. Perhaps the authority doesn’t understand that (so much context gets lost in isolated quotation that it is hard to tell). But he doesn’t say “there will be *no* interface.” He says “you won’t have to *worry* about the interface.” It’s a fair bet he’s not talking about the if-you-hate-voice-systems-say-“yes,” menu-driven, hierarchical, voice interaction — which you *do* have to worry about. He is talking about talk, about speaking, about conversation.

What voice interfaces require, as Hayes and Reddy put it very early on in the development of these systems, is the capacity for “graceful interaction” (1983); conversation is the model for graceful interaction using voice.

Talking to Machines

There is evidence that people’s communication with computers differs [from] their communication with humans.

— Niels Ole Bernsen, Hans Dybkjær, and Laila Dybkjær

Talk is not all the same. It occurs within *genres*, just like text (hand-scribbled notes, email, junk mail, novels) and television (sitcoms, documentaries, award shows, dramas) occur within genres. For that matter, locomotion (walking, running, driving, sitting in a rickshaw) and shopping (for hardware, for groceries, for lingerie) have distinct genres. Activity has contexts, governing conditions, strategies; and talking is a hugely distributed

activity, broaching many contexts, conditions, and strategies. Hardware-store talk is different from lingerie-store talk, which is different from ordering a hamburger or greeting a neighbor. Moreover (to admit just a little bit of complexity before retreating again to simplicity) these categories can overlap and interpenetrate: it may be that our neighbor sells hamburgers, or that we are shopping in a department store for a hammer and a Valentine's Day gift at the same time.

Looking only at the narrow matter of who-talks-when, genres of talk can

be viewed as a continuum ranging from the relatively unconstrained turn-taking of mundane conversation, through various levels of formality, to ceremonial occasions in which not only who speaks and in what order, but also what they will say, are pre-arranged (for instance, wedding ceremonies).

(Hutchby and Woffit, 1998: 147; they are paraphrasing the groundbreaking but less-succinct Sacks, Schegloff, and Jefferson, 1974.)

One might also locate genres of talk on other continua, defined by, say, content (hardware, lingerie, hockey), context (store, phone, street corner), or purpose (commercial transaction, information exchange, social maintenance). The variables are extensive. And that's the point: when we talk about talk we have to (1) remain sensitive to genre, and (2) remember the variables.

The subject of this book (talk with machines) is an emergent genre. More accurately, it is a range of emergent genres. This subject has been explored seriously for over forty years, and imagined for hundreds more, but it is only now coming into existence. Only now are the contexts and conditions, conventions and strategies, being fashioned at both ends of the telephone. We don't yet know how people prefer to talk to machines. We definitely don't know how they will be talking to them in twenty years. But there are plenty of hints, lots of studies, and we all have hunches. My strongest hunch (but certainly not *only* mine), and the belief that drives this book, is that people will want to piggy back their speech-dealings with machines on the cognitive and social evolution of language; that they will, for instance, prefer continuous speech to isolated words; that they will want to manage their inputs, in response to the machine's outputs, much the way they manage conversational turns with other people; that they will be happier with a system which allows them to participate actively rather than one that leads them through a series of navigational obstacles.

This belief is *not* that people will treat talking machines exactly the way they treat talking organisms. On average, there will surely be less concern for social graces, along with far more willingness to issue direct commands, to interrupt, and to be generally more abrupt. For instance, people often use prerequests to get attention and set up an information query, but utterances like "Can I ask you something?" are not likely to be commonly directed to a voice system; or, if they are, they would far more likely be simple, direct questions (rather than prerequests) by a user who wants to check on the interactive possibilities of the system.

On the other hand, there will surely be more accommodation made for conceptual and performance errors; people speak differently to children, foreigners, and persons with cognitive deficits than they do to adults who are cognitively adept and fluent in their own language. They won't have the same kindness and tolerance, surely, towards the voice-interface front ends to powerful multinational corporations that they might show towards a mentally challenged young woman asking for directions. But people do have the capacity to alter their articulations and expectations as a function of their partners-in-talk, and this capacity is called upon by voice interfaces.

But these actual and potential differences do not argue, as some people take them to, that the path to better voice interfaces should not run through the wide territory of natural speech interaction research. On the surface, it might seem to make more sense to study how people interact with voice systems now. That research is clearly important, but it is nowhere near sufficient, and used exclusively will only result in dull, unimaginative systems. Since (1) voice systems have manifold technological limitations and design flaws, and (2) speakers have a remarkable capacity to adapt to language behaviors, even primitive and bizarre language behaviors, studying current voice system interactions is of quite limited use in understanding what the optimal design principles are for voice interfaces. That's like studying how people shuffle around in manacles in order to understand the optimal design principles for running shoes.

We don't know the specifics, because these interfaces are so new. But to generalize from how people interacted via speech with computers as recently as five years ago to how they will interact with a system built in the next five years is a highly questionable enterprise. To generalize from how they speak to each other, however, while not without its risks, provides a more promising foundation.

The idea is not to build machines that can converse fluently on 19th century aesthetics or the implications of iridium layers for paleobiology, though such machines will surely come in this century. The idea is simply to build machines that allow people to exploit their naturally cultivated linguistic and conversational skills, rather than developing an entirely new repertoire of language behaviors because the interface designers think it is perfectly reasonable for a speech system to talk like this:

TOOT Hi. This is AT&T Amtrak Schedule System. This is TOOT. How may I help you?

Caller I'd like to get a train from Philadelphia to New York.

TOOT I heard you say "go to New York from Philadelphia." When do you want to leave?

Caller Sunday.

TOOT I heard you say "leave on Sunday." What time do you want to leave?

Caller 10:30 p.m.

TOOT I heard you say “around 10:30 p.m.” Do you want me to find the trains from Philadelphia to New York on Sunday around 10:30 p.m. now?

Caller Yes.

TOOT I am going to get the train schedule for you . . .

(Walker, Kamm, and Litman, 2000: 6)

Never mind the bizarre, and bizarrely iterated, phrase “I heard you say ‘X.’” In every single case that “X” is wrong, it is not what the caller, in fact, said. The caller in this exchange is remarkably cooperative, but TOOT is not; in fact, it violates a well-known conversational maxim such that providing more information than is required during your turn is surly. A humanly designed speech system, one that operated in a way that learns from how humans engage in dialogue, would talk more like this:

Caller Sunday.

TOOT-2 Sunday? What time?

There is confirmation here, which speech systems require far more often than humans; the hypothetical TOOT-2 does not act just like a person. But it doesn't fall into rote confirmation, either. And TOOT-2 takes the next conversational move (seeking a departure time) efficiently.³ The systems don't have to be human, just — to use Jef Raskin's beautiful design term — humane.

What Isn't in This Book

You can't have everything. Where would you keep it?

— Steven Wright

This book is largely unconcerned with the mechanical specifics of speech systems, or their implementational languages. I follow the advice of Michael Norman and Peter Thomas about the application of conversational human factors principles to speech-system

3: I am acutely aware of the arrogance involved in playing armchair designer, as I do here and elsewhere in the book — sitting back and proposing changes on paper to a real, working system that inventive and hardworking people have sweated to develop — and, in fact, the current Amtrak speech system (1-800-USA-RAIL) is somewhat less dogmatic than the original TOOT. But the progressive road for interaction design, as everyone who works in the field understands fully, is iterative, and I am just trying to use this iterative spirit to investigate design possibilities and to push voice interface development forward.

development, which they say “should be *concerned with* technology, but must be *independent of* technology” (1990: 55).

There is a little bit here and there about recognition and synthesis and natural language understanding, but only at the most basic level, and only from the design perspective (to understand, for instance, why recognizers are so error prone, or what characteristics of synthetic speech can be altered to influence the perception of emotion). There are already good books on speech technologies, by more knowledgeable people. (As we go to press, the best ones in my estimate are Robert Rodman’s [1999] *Computer speech technology*, for elegance, clarity, and patient explanation; Douglas O’Shaughnessy’s [2000] second edition of *Speech communications: Human and machine*, for comprehensiveness and detail; and Christopher Schmandt’s [1994] *Voice communication with computers: Conversational systems*, for careful attention to the constraints these technologies place on voice interaction design. The field is developing quite rapidly, however, so there may be more up-to-date books within a year or two of the publication of this book, *Voice Interaction Design* — perhaps even new editions of Rodman, O’Shaughnessy, or Schmandt.)

The speed of technological advances, in fact, is the principal reason for this book’s indifference to speech technologies. The advances in signal processing algorithms, memory, and brute power have far outstripped design in speech-system research. This book is an attempt to push design.

While there is an undeniable developmental reciprocity between technology and design principles, the latter are a good deal more stable and always need to take the driver’s seat. Those of us who were around for the frustrating early years of graphic interfaces remember a lesson that we can take for speech-system development — from the many awkwardly designed systems that sprang up and died while technology (the applications) ruled supreme. When graphics experts and human factors specialists began to develop controlling interests in graphic interface design, things changed very much for the better. Speech system development currently needs more language experts and human factors specialists than coders. This book aims to help push the field in that direction.

My Approach

Contrary to popular opinion in at least some circles, applications of speech technology don’t generally fail because a speech recognizer’s accuracy is 93% instead of 97%. . . . They fail because human factors concerns are not addressed.

— Daryle Gardner-Bonneau

I have a broader conception of voice interaction design than many current practitioners, who see voice interaction design and development as challenging and important work, but still as something of an add-on to the actual, true, and real engineering object, the speech system — that complex of circuits and code that defines recognition, parsing,

language-modeling, and speech synthesis. In this view, a common one, “the interface” is isolated to one module, often called the dialogue manager. In my view of human–computer interaction (by speech in specific, but by any modality in general), there is no such segregation. A voice interface is the sum of all human-factors concerns in the design and development of a speech system (including, but not limited to, “dialogue management”). My arguments to this end are bound to be controversial in some quarters; they have already rubbed some practitioners the wrong way.

I not only insist at every opportunity on the primacy of human factors concerns in speech system development — exactly what you would expect, I would hope, from someone writing on user interface design — I insist on the necessity of a thorough grounding in the specific, rich human factors research base on how people use language interactively. This base has been developing for millennia, and our travels sometimes take us back to Ancient Greece, but it gathered incredible momentum in the last half of the 20th century, in a range of overlapping fields, chiefly linguistics, philosophy, sociology, and psychology. I also insist that heavyweight resources — both technological and human — be put behind the application of this research to voice interface development. This insistence especially has been called extravagant, excessive, idealistic — not by everybody, but by a noteworthy contingent of old-guard speech-system folk.

Yet, these arguments really only come down to the low-level application of common sense. Language is a fundamentally human instrument, with a distinct and complex character. Its composition and use is certainly related to general cognitive principles (which I also advocate voice interface design respect, of course), but it also has a wide range of unique elements, relations, and protocols; moreover, even general-purpose cognitive principles have somewhat distinct effects in the realm of linguistic interaction. Attention, for instance, and salience, have quite specific realizations in language.

Designing speech-based interactive systems with only a loose understanding of this character dooms those systems to mediocrity, if not outright failure. And thorough understanding does not come cheaply. Understanding the nature of linguistic interaction in a theoretical way takes energy and time. But understanding it in an applied way takes more. It takes energy and time too, but also money, machines, and methodology.

To understand the way a community uses language to accomplish a set of goals, like booking tickets or ordering pizzas or doing some banking, requires closely studying that language domain. It will have its own terminology, its own utterance structures, and its own interactive patterns. These need to be collected into two interdependent databases — a general one, called a *corpus*, which houses and organizes a wide sample of language use, especially dialogues; and a specific one, called a *lexicon*, which houses and organizes the characteristic words of that domain. Building and then using such databases requires people and computers. My advocacy of them (and more particularly, of corpus linguistics) for voice interface design is a hallmark of the extravagance for which my approach has been criticized.

But these tools, or tools very much like them, are mainstays of the other areas of speech-system development, and all I am ultimately advocating is a resource allocation model that puts the same effort — technological and human — behind interface development as behind recognition, parsing, and language understanding. That seems perfectly reasonable to me. Moreover, since, as Roni Rosenfeld notes, conversational systems “require a lengthy development phase,” anyway, “which is data and labor intensive” (quoted in Green, 2001), it is deeply misguided not to foreground human-factors concerns throughout that process. Leaving user issues to the end doesn’t work with graphic interfaces; leaving them until the end with a system as dependent on user dispositions as an interactive speech application is an even worse idea. The speech-system development cycle, as I propose in the book, should be a voice interface development cycle.

I also argue for resource-heavy testing, for the use of other digital tools, like digital design specifications, the use of knowledge-representation scripts as the basis of the interaction model, and the model of the voice interface as an expert system. From the human-factors language research base, I draw on the findings of lexicography, conversation analysis, pragmatics, computational linguistics, and the collaborative-action framework of social psychology, as well as a range of more local disciplines, like phonetics, syntax, and semantics. Some of the components of this *mélange* are more accepted than others — but collectively they nowhere receive the kind of attention I give them here.

My terminology, too, is occasionally nonstandard — perhaps most egregiously for many traditional speech-system people in my use of the word *vocabulary* for what some of them call a *dictionary*, and most call a *grammar*; that is, for the storehouse of interrelated acoustic models that defines the “sublanguage” of the interaction: what the users can say, what the system can hear. But I have also adopted or developed words from a range of disciplines to cover the overlaps and amalgams that are necessary when dealing with such a large and diverse body of research. In all cases, I’ve done this solely in the interest of conceptual clarity. Individuals who have investments in one bag of terminology or another may not take kindly to the way I have altered or ignored their terms, but I hope that I have served the design community’s best interests.

And, while there is no shortage of opinions and recommendations in this book, I have avoided a formulaic connect-the-dots approach that would falsely suggest that you just have to connect up X, Y, and Z if you want a great voice interface, only X and Y for a good one. Such design books, and there are many, trivialize the richly creative, collaborative process of crafting intricate, interactive artifacts for human use. And they are misleading. “Design is complex,” as Deborah Mayhew, among many others, has noted, “and there is no cookbook approach that can rely on general principles and guidelines alone” (Mayhew, 1999: 4).

The point of this book, simply put, is to provide voice interaction designers with the knowledge and the strategies to craft language-using applications that behave the way a language user expects other language users to behave. But that knowledge and those strategies do not provide a prefabricated speech system.

The Rest of This Book

[In] the fusion of art and technology we call interface design . . . the inventors and practitioners have blurred into one holistic unit, like a science lab hosting a creative writing seminar.

— Steven Johnson

A major reason that good graphic interfaces (integrated with useful tools) are so effective is that they capitalize on the contextually embedded, your-turn-then-my-turn, mutually reinforcing, collaborative strategies of human-human interaction. Using computers in the 21st century is far closer to the way we interact conversationally with other people than it is to the way we use a shovel or a bicycle or a chain saw. All user interfaces are dialogic — turn-by-turn communicative exchanges of information and intention, more-or-less similar to the dialogues between humans. But with non-voice interfaces this dialogic quality remains metaphorical.

Voice interfaces are literally dialogues: two agents communicating in one coherent, spoken discourse. So, while the principles of discourse have lessons to offer the design of all human-computer interactions, those lessons are much more direct and solid and authentic for voice interface design.

There are still figurative elements to all this. The “agent” at one end of the line does not have the cognitive and social capacities of the agent at the other end. Notions like intention and personality and even communication can be used at the machine end of the line only under an explicit fiction, where the voice represents a system, a company, its designers and developers, the way a particular main window (with its menus, bars, and palettes) represents Illustrator, Adobe, Rick Boyce, Jeff Bradley, Paul George, and so on. But because the system, institution, and people are condensed into a voice, these figurative aspects are fading, and it may not be long before terms like “ticket agent” and “travel agent” primarily denote synthetic creatures.

The extent to which the figurative aspects recede, and the speed with which they do so, will depend on one thing, and one thing only: the facility these agents can acquire with spoken language. That facility is the focus of the rest of the book.

First, in **Part I: Talk**, I outline some of the features and principles of speech, from acoustic wave form to discourse. The material is unabashedly opportunistic, and for the most part oblivious to theoretical nuance. There have been, for instance, a good many exchanges among representatives of speech-act theory (philosophers and linguists, mostly), conversational analysis (sociologists), and discourse analysis (sociologists again, with literary theorists and refugee linguists thrown in). Some exchanges are little more than embarrassing turf wars, some are serious attempts at syntheses, or outright absorptions, and some are just dismissive hand-waving. Some are useful, some trivial. I have simply plundered this literature for what speech applications need, and ignored the counter-arguments about,

say, whether speech-act theory makes sense in the context of a conversation, because (for instance) strict versions of the theory require seven components of illocutionary force, none of which determines the appropriate conditions for a reply to that act, and conversations depend on understanding reply conditions. If I can jerry-rig two approaches together sufficiently to generate something useful for voice interaction design, that's good enough for me. And there is more jerry-rigging going on in Part I than trying to reconcile speech-act theory and conversational analysis and text linguistics. The whole chapter is a stew of results from fields and approaches that sometimes view each other with antagonism. Tough. Let them fight it out. We'll take what we can use.

The chapters in this section are:

Speech, an introduction to the section.

Sound and Meaning, an overview of the mechanics of language, follows the conventional (and fruitful) approach of four specific focal ranges: the phonological, which attends to sound; the lexical, which attends to the words built out of those sounds, when they are linked to meanings; the syntactic, which attends to how words travel in certain acceptable packs (those packs usually labeled “grammatical”); and the semantic, which attends to the way those packs make composite meanings (“Bart poked Lisa” and “Lisa poked Bart” have all the same elements, but different composite meanings). The chapter also briefly takes up the more distributed linguistic notion of prosody, the rhythms that give speech its life, and sketches out the processes of speech synthesis.

Doing Things with Words, a survey of the impact of context on language use, draws mostly on the traditions of ordinary language philosophy, which opens up the focal range even further to include circumstances and purposes (“Fire!” is a very different utterance in a crowded theatre than on the battlefield). Chief among the circumstances and purposes considered in this chapter are the ones that govern conversational exchange.

Conversation, a review of the principles guiding what some theorists call talk-in-interaction, continues to draw on ordinary language philosophy, particularly for the way in which it reveals that speech is a form of action (“yes,” for instance, doesn't just signal agreement; it might initiate, or finalize, a purchase). The organizing insights of the chapter are from a branch of sociology called *conversation analysis*, and a branch of psychology which doesn't have a snappy label but which takes a perspective on talk that articulates the single most important attitude for voice interaction design, almost a mantra: that talking is a species of *collaborative action*.

Glue, an overview of the networks of reference and relations that bind dialogues (and all discourses) together. The words, utterances, turns, and exchanges consolidate into dialogues because they invoke the same entities and they support each other, on the two elemental levels of language, content and form. Those networks effect the coherence and the cohesiveness of the dialogue. Coherence is a function of the conceptual networks, cohesion of the formal networks.

Diction, borrowing a somewhat quaint-sounding word from rhetorical theory, centers on the single most critical element of voice interface design, choosing the optimal word for

a given audience, purpose, and context. We draw most fully here on the work of corpus lexicographers, people who figure out how words function by looking very broadly at how they are used.

In the second section, **Part II: Design**, I outline more specifically, and in greater detail, the processes, principles, and practices of crafting voice interfaces. The natural model for comparison is the design of graphic interfaces, which transfers very well in one way, and quite poorly in another. The overall graphic interface design *process* carries over very well: up-front task and user analysis, iterative development with lots of testing, and situated design. There are differences in the development cycle — in the depth of domain analysis, especially the use of natural dialogue studies, and in the importance of Wizard of Oz studies — but the most significant differences are in the specific crafting of the interface. The design medium, spoken language, means a much different palette of tools to work with, and a much different set of constraints to work under.

The chapters in this section are:

Crafting Voice Interfaces, an introduction to the section and a treatment of several basic issues, starts with the first question to ask, at the outset of the design process, whether or not the target service is appropriate for voicing. Of particular concern, because there is both so much potential and so much hype, is voicing web sites, a highly visual medium which often has text that is poorly suited for voicing. We also take up two sets of design contaminants — the attempt to apply graphic design sensibilities to a verbal medium, and the use of menus in a speech system. The chapter rounds to a discussion of the guiding sensibility of this book generally, and the design section specifically, the pursuit of habitability, the property of a discourse model to provide users the room they need to achieve their goals in linguistic comfort.

The Team and the Process presents a roll call of the job functions necessary to develop a habitable voice interface, and the iterative process in which they can do it. The roll call is controversial, chiefly in its overall size and its inclusion of a Lexicographer. But all the roles are unquestionably necessary for the development, even if several of them are combined in one individual: the leader is an Interaction Architect, the Lexicographer is responsible for the research necessary to build the discourse model, Dialogue Writers put the speech system's utterances together, the Soundscape Designer creates the nonspeech audio, the Quality Assurance Prime oversees the development process and enforces the milestones, and the Usability Prime guides the heavy user testing necessary for voice interface development; Subject-matter and Speech-technology Experts are also important team members. The development process this team follows is the familiar spiral model of iterative expansion, from prototyping through release.

Users, Tasks, an outline of data-gathering considerations and strategies for understanding the users and their tasks, starts with suggestions on how to capitalize on whatever existing data there might be on the people and processes in the target domain for the voice service, but spends most of its time outlining primary research techniques for observing and interviewing the users, as well as charting and analyzing their tasks.

Building the Discourse Model revisits habitability in terms of the goals necessary to achieve a coded understanding of the register that defines the voice interface, in terms of lexical, syntactic, and functional criteria. This chapter advocates the use of natural dialogue studies and strongly recommends the adoption of two powerful design instruments in carrying out and profiting from those studies — a dialogue-rich, register-specific corpus, and a digital lexicon.

Agents is a discussion of the principal considerations in designing the character(s) of a voice interface. There is, unaccountably in my view, a controversy about whether voice interface agents should be human-like or not (*anthropomorphic* is the preferred term of the controversy, though I prefer *personified*). I review the controversy, mostly to demonstrate that voice interface agents, by their very linguistic nature, are already personifications, and the design questions concern how one comes to grips with that fact. The chapter outlines the best ways to come to grips: issues of branding, aesthetics, personality, gender, emotion, and ethical character are all investigated, as well as the reasons for recorded-voiced and/or synthetic-voiced agents, and the reasons for using a single or using multiple agents in the interface.

Dialogue Matters is largely an outline of strategies and concerns with respect to communicative slippages. Things always go wrong with speech systems — in part because of the inherent fragility of the technology, in part because of the immense difficulty of working with a medium as slippery as spoken discourse — and this chapter focuses on how to minimize those slippages. It offers a taxonomy of errors and slippages. It recommends preventative measures such as effective prompting, careful vocabulary building, and certain resolution strategies. It recommends how to fix the slippages that do occur, through strategies that guide the user in the repair process. And it treats other significant dialogue matters related to prevention and repair — managing initiative, tapering and expanding system output, as well as issues of working with preexisting text, especially from graphic sources, and concerns that arise because of the legacy often inherited by speech systems that replace human or keypad interactions.

Scripting is the development of a detailed design specification, with two principal components, writing the dialogue and planning the call flow, both of which this chapter charts out thoroughly. It develops the idea that a conversational voice interface is a form of expert system and begins with a conventional knowledge script for a take-out/delivery order service, elaborating that script in the direction of dialogue acts and conversational turns, on the one hand, and of an interactive architecture on the other. This chapter also advocates the adoption of a digital design spec, rather than (just) a paper-based document.

Iterative Evaluation, an account of the testing procedures needed throughout the development cycle, pays particularly detailed attention to a method highly suited to voice-interface development, Wizard of Oz testing. Oz testing has a human (the Wizard) simulating a speech system, which allows for the early and rapid testing of design ideas. The human, and the necessary support resources (computers, other humans, possibly a

vocoder), are unique to this methodology, but otherwise the process is of a piece with usability testing, so I spend comparatively less attention on such testing, but I also review heuristic evaluation methods, and beta-testing, field studies, and what I call the pluralistic talkthrough.

Conclusion — Pursuing Habitability, briefly wraps up the themes of the book.

Glossary; there is also a comprehensive glossary of relevant terms from linguistics, philosophy, sociology, telephony, human-computer interaction, and speech system research.

Summary

The need for conversational agents has become acute with the widespread use of personal machines with which to communicate.

— Yorick Wilks

“Today, you have to understand the systems,” W.S. “Ozzie” Osborne (general manager of IBM’s Speech Systems division) said at the turn of the century, “but soon the systems will understand you. That’s the transition over the next couple of years, from you learning machines to machines learning you” (quoted in Hellweg, 1999). That’s what this book is about: helping the designers of the machines build ones that better understand and accommodate people’s speech behaviors.

In this chapter, I have outlined the various user interfaces relevant to the evolution and design of voice interfaces, including a general description of voice interfaces themselves. I have also argued for the importance of speech as an interaction modality in a range of circumstances — chiefly eyes-and/or-hands-busy circumstances — and for the centrality of human–human conversation in voice interaction design. I have sketched out the main components of this book. And I have articulated a vision of the driving importance of human-factors research to the design of voice interfaces, as well as conceiving that research very broadly, to encompass the work done in a range of (sometimes mutually hostile) disciplines on the fundamentally human phenomenon of interactive language use.

