WORKING PAPER
ALFRED P. SLOAN SCHOOL OF MANAGEMENT

The Design Spaces of Audio Menus and Forms
Paul Resnick*
Robert A. Virzi**
MIT Sloan School Working Paper #3633-93
MIT Center for Coordination Science Technical Report #159
November, 1993

MASSACHUSETTS
INSTITUTE OF TECHNOLOGY
50 MEMORIAL DRIVE
CAMBRIDGE, MASSACHUSETTS 02139
The Design Spaces of Audio Menus and Forms
Paul Resnick*
Robert A. Virzi**
MIT Sloan School Working Paper #3633-93
MIT Center for Coordination Science Technical Report #159
November, 1993

Abstract
Menus and forms are important dialogue structures in telephone-based interactive voice response and other audio applications. There is a surprising lack of diversity, however, in the interaction styles they employ. This article presents design spaces for audio menu and form styles. The key idea is to break recordings and actions into parts. The methods of recombining the parts are the dimensions of the design spaces. Twelve alternative menu styles and five form styles illustrate some of the recombination possibilities. Choices on each of the design dimensions affect user interactions, the sound and feel, in predictable ways. The best style will depend on the experience levels of an application's users.

*MIT Room E53-325
Cambridge, MA 02139
presnick@mit.edu

**GTE Laboratories Incorporated
Waltham, MA 02251
rvirzi@gte.com
1. Introduction

Digital storage and processing of audio have opened new possibilities for speech-based applications. There is already a large and growing market for telephone-based voice mail and interactive voice response services. With the advent of personal digital assistants and the integration of audio into desktop computing, speech is also likely to gain importance for eyes-busy applications, for personal communications, and for records of conversations.

Compared to visual presentation of information, speech output is slow, serial, and provides no short term memory aids [Halstead-Nussloch 1989; Schmandt In press]. Good readers can read faster than they can listen. Two technologies are available, however, that can aid listeners. The first is to play speech back faster than it was recorded. An increasing number of voice mail systems offer accelerated playback, usually without pitch distortion. Some digital signal processing mechanisms allow a factor of two speedup while still retaining intelligibility [Arons 1992; Kato and Hosoya 1993].

The second, and more important technology, is random access. It takes time to fast forward a conventional audio cassette tape, but it takes virtually no time to jump to a different part of a digitally stored recording. Meaningful subdivision of recordings, together with user control over jumps between those parts, allows listeners to skip some parts of a recording entirely. Elsewhere, we have described as *skip and scan* those audio interfaces that allow users to scan a recording by skipping frequently [Resnick and Virzi 1992; Virzi, et al. ]. [Arons 1993] explores playback controls that affect both speed of playback and skips between segments.

To exploit random access, a designer needs to identify meaningful segments in recordings. One source of segmentation is pre-defined structure, such as the separate
entry blanks in a form. The person recording can also indicate segment boundaries [Degen, et al. 1992; Gould and Boies 1983; Stifelman, et al. 1993]. In some cases, a computer can infer segments after the fact from acoustic properties of a recording, such as turn taking between speakers [Hindus and Schmandt 1993].

This article describes what user control, through random access, can do for two common audio dialogue structures, menus and forms. A menu allows selection of one or more options from a pre-defined set. A form allows entry of a collection of related pieces of information. Audio menus and forms present options and instructions through spoken voice and allow user input either through buttons or speech.

Many menu and form styles are possible. The optimal choice is likely to vary between applications. There is a surprising lack of diversity, however, in the styles commonly employed. We organize the design space to help system builders, interface designers, and human factors researchers explore the possibilities. For system builders, the fundamental strategy is to break dialogue components into smaller pieces, then reconstruct them in novel ways. We show that many styles can be constructed from the same primitive parts, by varying a few features of how they are glued together. For interface designers, we discuss how variations in those construction features affect user interaction, the sound and feel. The preferred sound and feel will differ among users depending on their experience with an interaction style and a particular application.

All our examples assume interaction over a telephone. The analysis applies, however, to any audio presentation of menus and forms, over the phone or with some other device such as a Personal Digital Assistant. As we will argue in the conclusion, the analysis is even relevant to limited bandwidth visual output devices, such as 20 character by two-line LCDs, since such devices have the same temporal presentation constraint as audio output.
The critical factor that distinguishes our analysis from analyses of most visual menus and forms is the temporal presentation of information. Some analyses of visual menus assume that users consider the options one at a time [Lee and MacGregor 1985; Paap and Roske-Hofstrand 1986], while others assume a more flexible process [Card 1982; Landauer and Nachbar 1985], leading to different conclusions about the optimal breadth or depth of menu hierarchies [Kiger 1984; Miller 1981]. [Norman 1991] summarizes much of this literature. All of these analyses, however, assume simultaneous presentation of the entire menu: a user shifts attention between items by shifting eye gaze.

Two experiments explored visual menu styles that were artificially restricted to temporal presentation of items [MacGregor, et al. 1986; Pierce, et al. 1992]. The screen displayed only one menu item at a time; users controlled when to move to the next item with keypad input. The models of human search processes developed from those experiments will likely apply to some audio menu styles but not others because not all audio menu styles give users control over when to hear the next item.

Our examples all assume touch-tones for input, with the buttons referred to by number (0-9) or symbol (* and #). Keys could also be labeled by letters, however, so that users could enter a letter sequence or word [Davis 1991; Detweiler, et al. 1990; Fast and Ballantine 1988; Marics 1990] to initiate actions. Buttons on a hand-held device would lead to similar interactions. We briefly mention input by speech recognition in those situations where it could offer significant advantages over buttons.

Sections 2 and 3 elaborate the system and user perspectives on the design of audio dialogues, drawing on menu styles to illustrate. Sections 4 and 5 apply these perspectives to describe and analyze design spaces for menus and forms. In both cases, the system perspective summarizes existing styles and suggests new ones. The user perspective suggests when particular styles will be most appropriate.
2. Overview: the System Perspective

The system perspective follows a traditional engineering approach. Divide a dialogue mechanism into its constituent parts and recombine the parts in novel ways. In this case, there are two kinds of constituent parts. The first are voice recordings. The second are actions such as marking a particular item in a menu or adding a value to a particular entry blank in a form. Four design dimensions govern the recombination of these parts:

1) **Action combinations**: Which component actions and combinations can users initiate?

2) **Action Distribution**: From which recordings are actions available?

3) **User Inaction**: What effect does user inaction (a timeout) have from each of the recordings?

4) **User Initiated Movement**: What transitions between the recordings can users initiate?

We introduce these ideas by applying them to the most popular implementation of audio menus. Section 4 will recursively subdivide the recording for each menu item. Section 5 will generate alternative form styles through the same technique of subdivision and recombination.

The predominant implementation of audio menus currently is as a single recording that describes all the options sequentially. Any time during playback of the recording, a caller can press a number associated with an option to select it. Example 1 presents a sample interaction with such a menu. We first consider division of the recording and then division of the selection actions.
Welcome to the ABC Bank's bank-by-phone.

For account balances, press 1:
To transfer money between accounts, press 2;
For mortgage rates, press 3.
To open a new account, press 4.

This menu will now repeat. Make your selection at any time.

[presess 4]

Example 1: the standard menu style. System prompts appear in plain text, while user actions are bracketed. The caret symbol, ^, inserted in the prompt, indicates when the key press is made. That is, the caller presses 4 before hearing that the menu will repeat.

Figure 1: A diagram of the standard menu style implemented with separate recordings for the header, menu items, and footer. Boxes indicate recordings. When the system finishes playback of the recording in the current box, it begins playing the box just below it. Arrows inherit from the outside in. Thus, the selection actions are available from the header box, the item boxes, and the footer box, even though they are shown only once for the enclosing box.

The recording can be divided into six parts: (1) an introductory header, (2) - (5) descriptions of each of the menu items, and (6) a concluding footer. Some menu styles may omit the header or the footer. Other divisions are possible, but this one seems particularly natural. Each selection action can also be divided into two component actions, one that marks a preference for an option and one that terminates interaction with the menu.

Appropriate choices on the four design dimensions can duplicate the interaction style of Example 1. First, keep the selection action as a composite of marking an option and terminating interaction with the menu. Second, make all the selection actions available from all the recordings. Third, automatically transition to the next part when users are
passive. Fourth, do not allow any explicit user transitions between the parts. Figure 1 summarizes these choices in a notation that we use throughout the paper.

Consider how a user interaction would proceed, given those design choices. The system begins by playing the first part, the header. If the user does nothing, the system automatically transitions to playing the description of the first option, then the second, and so on. At any time, the user can press a numbered button to select any of the options. Moreover, there is nothing else the user can do but wait or select. There is no way, for example, to jump from the header to the description of the last option.

Dividing a recording would not be very interesting if it only led to duplication of the original interaction style. Other choices on the design dimensions, however, lead to alternative interaction styles. Consider some other possibilities.

First, what if separate marking and termination actions were provided? The standard menu style combined these into a single select operation. If they were separate, however, a user could mark several items before terminating, thus selecting several items from the menu.

Second, how are the selection actions distributed among the parts? In the standard menu style, all of the parts inherited all of the selection actions. Another possibility, however, is to distribute the actions positionally, so that a menu item can only be selected while listening to the recording that describes it. A single button, then, could be used to select the current item rather than providing separate selection buttons for each of the items.

Third, what effect does user inaction have? In standard menus, inaction caused an automatic transition to the next part. Another possibility is for inaction to cause repetition of the current recording. A third possibility is for inaction to cause selection. Users would explicitly reject each option; doing nothing would cause the current item to be selected.
Fourth, what transitions can users initiate between the parts? Figure 1 did not allow any explicit user navigation among the parts. Another possibility is to provide a “next” button that advances from one part to the next. For example, a user could interrupt the description of one menu item by pressing the “next” button. The system would immediately start playing back the next item. Other explicit transitions are also possible, such as jumping ahead by several options, or restarting the menu.

2.1 An Alternative Style

Consider an alternative menu style, shown in Example 2. Since it uses only two buttons, we call it and its variants 2-button styles. One button advances to the next item. The other button selects the current item. In terms of the design choices, it still uses a composite select button, but it distributes the actions positionally, timeouts cause repetition of the current item, and it provides an explicit “next” button to transition to the next item. Figure 2 summarizes the interaction style.
Welcome to XYZBank’s bank-by-phone. To hear the first option, press 3. ^
+ [presses 3]

Account balances. To select this option, press 1. For the next option, press 3. ^
+ [presses 3]

Transfers between ^ accounts. To select this option, press 1. For the next option, press 3.  
+ [presses 3, interrupting prompt]  

Mortgage rates. ^ To select this option, press 1. For the next option, press 3.  
+ [presses 3, interrupting prompt again]  

Open a new account. ^ To select this option, press 1.  
+ [presses 1, interrupting prompt again]

Example 2: A sample 2-button menu dialogue.

Figure 2: The 2-button menu style. When there is no adjacent box, it replays the contents of the current box. That is, the current item will keep replaying until the user presses a button.

2.2 A Simple Subspace

Not all combinations of design choices yield plausible menu styles, but many do. Figure 3 summarizes a subspace defined by some of the possible choices. The rows indicate three choices for transitions from one part to the next: automatic (timeout) transitions, explicit user actions, or both. The columns indicate the two choices for distributing selection actions among the parts: absolute or positional. The entire subspace assumes a composite selection action, so that users can select only one item from a menu.
The two styles above fill two of the cells. Absolute selection together with automatic transitions between parts defines the standard menu style. Positional selection along with explicit transitions defines the 2-button style.

The other four cells mix and match these features. The temporal menu style [Schmandt in press] uses only one button. The listener waits while the computer recites the options and presses the select button upon hearing the desired option. The *temporal with skips* style adds an explicit skip ahead button to temporal menus, so that a listener can either wait through the recitation of the options, or press a button to skip through them. The *standard with skips* style works just like standard menus, but provides in addition a button, say #, that a listener can press any time to skip ahead to the next option. The *stepped numeric* style removes the automatic advance, so that a listener can only move on to the next option by pressing the skip key.

<table>
<thead>
<tr>
<th>How to advance</th>
<th>Absolute Numeric</th>
<th>Positional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeouts</td>
<td>Standard</td>
<td>Temporal</td>
</tr>
<tr>
<td>Skip key</td>
<td>Stepped numeric</td>
<td>2-button</td>
</tr>
<tr>
<td>Timeouts and Skip key</td>
<td>Standard with skips</td>
<td>Temporal with skips</td>
</tr>
</tbody>
</table>

Figure 3: The subspace defined by the dimensions of how users select items and how they advance from one item to the next.
2.3 Dividing the Selection Action

Any of the styles in the subspace above could be modified to use separate marking and termination actions rather than a composite selection action. The positional multi-selector style is a variant on the 2-button style. Example 3 shows a sample interaction.

Welcome to XYZBank's bank-by-phone. To hear the first option, press 3. *
(presses 3)

Account balances. To select this option, press 1. For the next option, press 3. If you're done making selections, press #. *
(presses 1, interrupting prompt)

Transfers between accounts. To select... *
(presses 1, interrupting prompt)
[Beep] Transfers between accounts. To deselect, press 1. For the next option, press 3. * If you're done...
(presses 3, interrupting prompt)

Mortgage rates. * To select... *
(presses 3, interrupting prompt)

Open a new account. * To select...
(presses 1, interrupting prompt)
[Beep] Open a new account. To deselect this option, press 1. For the next option, press 3. If you're done marking selections, press #
(presses #)

Example 3: the positional multi-selector style. The beep indicates that an option is marked for selection.

2.4 Summary of System Perspective

The system perspective defines a space of design choices. The design dimensions come from an analytic framework of subdividing recordings and actions and recombining them in novel ways. Four dimensions of choice govern the recombination: how primitive actions are combined into composite ones; whether actions are distributed positionally or some absolute mechanism is utilized; the effect of timeouts; and what commands are included for movement between recordings.

3 Overview: User Perspectives

From the user's perspective, some system level choices make little difference while others completely change the sound and feel of an interface. In addition, prompt wording
styles, which are immaterial from the system perspective, have a large impact from the user perspective. We discuss how users' experience levels may affect those interactions and then present general principles that relate system design choices and prompt wording styles to their effects on user interactions.

3.1 Experience Levels

All users are not alike. They have different goals, so they choose different actions. Moreover, they require different information to help them choose actions and figure out how to execute them. For example, users who already know how to select an option do not need to hear a prompt that says how to select it. It is the variation in user interests and skills that makes user control so desirable.

Figure 4 shows a graph of user types, defined by two variables. The first variable is users' familiarity with the mechanisms of the dialogue, how to initiate actions they have decided to take. For example, users familiar with the 2-button menu style know which button selects and which moves on to the next item. Such mechanism experts need no prompts to tell them how to execute actions. The second variable is users' familiarity with the contents of the dialogue, the information necessary to decide what action is appropriate. For example, frequent users of a voice mail application may be aware of all the options on the main menu, even though they may not know which keys are associated with some of them. Such content experts know which actions they want to take without hearing descriptions of them. Users who are both mechanism and content experts may not need to hear any voice recordings at all because they know both what actions are available and how to initiate them.
In general, repeated exposure will increase users' familiarity level with both the contents and mechanisms of a dialogue, but mechanism expertise tends to develop more rapidly. Regularities in the mechanism allow transfer of learning. For example, users of 2-button menus can predict the mechanism for selecting an option even if they have never selected that particular option before. Similarly, a user of a standard menu may be able to predict the number associated with the current option by adding one to the number associated with the previous option, because standard menus are usually numbered sequentially.

### 3.2 Design Considerations

A number of considerations govern how system level style choices will affect users with varying levels of expertise. Where appropriate, we cite empirical evidence from two user tests, reported elsewhere, that compared three menu styles [Resnick and Virzi 1992; Virzi, et al. 1992]. Two were the standard and standard with skips styles described above. The last was a variant of 2-button menus. It included a third button, to move back to the previous item. Since users almost never pressed it, we describe the experimental results.
as if 2-button menus were used. The data from the experiments are consistent with the design considerations but they do not provide conclusive proof. First, some of the outcome measures reported here were not part of the original experimental design. Second, the design considerations are stated generally for any dialogue structure, but the experimental data comes only from applying them to menus.

DC1) A combined action is simpler for users, while separate actions give them more flexibility.

The separation of selection from termination in a menu style permits users to make multiple selections or to change their minds about a single selection. A user who wants to select an option and then terminate can press two buttons in succession. This flexibility comes at the cost of increased complexity: when the two actions are combined, a user need learn only a single button.

DC2) Positional actions are easier to learn because they are independent of content.

The selection button is the same from any item in a 2-button menu, and from any item in another 2-button menu as well. With the absolute selection actions in standard menus, there are more mappings of actions to buttons that need to be learned since the selection button changes from item to item.

In the second of the two experiments, subjects required several exposures to a menu before they typed ahead numeric menu selections in the standard and standard with skips styles. 2-button menu users required much less practice before they stopped listening to the prompt for the single select button.

This design consideration suggests an advantage for speech input over keypad input when absolute actions are used. Command names may be easier to remember than button mappings. That is important when absolute actions are used since there are many mappings to remember. On the other hand, it takes longer to speak a command than press
a button, so there is a tradeoff between ease of remembering speech commands and ease of executing button presses.

DC3) Absolute actions are easier to execute than positional actions, once learned.

A user can type ahead a numeric menu selection with a single keystroke. Positional selection requires several keystrokes or waiting until the appropriate menu item plays back.

Surprisingly, the experiments did not provide clear evidence for this. After a fair bit of practice with the same menu tree and two repetitions of complete tasks, performance with all three styles was nearly indistinguishable. Careful analysis of the data logs indicates that users were typing ahead most but not all of their menu selections. We speculate that after even more practice, users would type ahead all of their selections, and then the numeric selection styles would have better performance than the 2-button style.

DC4) Automatic transitions help mechanism novices, but delay their acquisition of mechanism expertise.

A mechanism novice can hear all the parts more quickly if they are automatically played in sequence than if the caller has to learn what buttons invoke explicit transitions. On the other hand, omitting the automatic transitions forces callers to learn the explicit transitions. In the second experiment [Virzi, et al. 1992], initial performance was better on standard with skips menus, which include automatic transitions, than on 2-button menus. After practice on a few menus, however, users skipped more often in the 2-button style and made selections faster.

DC5) Automatic transitions, together with any position sensitive actions, create a 'moving target' problem for all users.

Both the temporal and temporal with skips styles combine automatic transitions with positional selection. If a user selects just as the system automatically transitions to the
next menu option, the wrong one may be selected. The ‘next’ button is also a positional
action, since it transitions to a different part depending on the current part. In either the
standard with skips or temporal with skips styles, suppose a user skips just as the system
automatically transitions from the first menu option to the second. The system will skip
ahead to the third item and the user may never hear about the second. As users become
more expert, and select or reject items sooner, the moving target problem becomes less
important.

DC6) When information relevant to a user is preceded in a recording by irrelevant
information, that user will pay a time penalty.

The designer can reduce the time penalties imposed by recordings that are irrelevant to
some users, in three ways: short recordings, explicit transitions, and ordering. Listening
to an irrelevant recording may not be so bad if it’s very short. Unfortunately, novice users
may make more errors if the descriptions of options and available actions are incomplete.
Many voice mail systems handle this tradeoff by including a novice mode with longer
prompts and an expert mode with shorter prompts.

Menu and form styles that support skipping provide another method for handling this
tradeoff. As callers gain expertise, they can skip the unnecessary portions of longer
prompts, listening to just enough to cue recognition of the entire prompts.

Yet a third technique is to order the recordings so that irrelevant information never
precedes relevant information. In general, this will not be possible because of differences
in user interest. Designers can approach the goal, however, by putting first the
information useful to the largest number of people.

DC7) Describing optional mechanisms degrades usability for mechanism novices
but encourages them to become experts.
Optional mechanisms are those that are helpful, but not necessary. For example, in the standard with skips menu style, a user who does not know about the skip button can still hear all the options and select one. If the instructions mention the availability of the skip key, novices who do not yet know how to use it will have to listen to the prompt, but will gain no benefit from it. The instructions, however, help the user to learn the skip key. If the instructions fail to mention the skip key, the novice user will perceive the menu style as identical to the standard style. This may be a reasonable style choice when there is some method for teaching about the skip key that is external to the interface itself. Such a method could be paper documentation, an explicit training session, or transfer of learning from some other interface.

In the 2-button menu style, on the other hand, neither button is optional. A listener must use them to hear the options and make a selection. A designer would have to be very confident of external learning to omit instructions for necessary mechanisms.

**DC8** Questions, commands, and pauses encourage users to take action right away. Users are more likely to wait for additional instructions when they hear a statement.

One study of standard menus [Engelbeck and Roberts 1990] measured how frequently users select an option immediately after hearing it (rather than waiting until the end of the menu). Subjects made fewer immediate selections with key-action wording of the menu items (“Press 2 to do something.”) than action-key wording (“To do something, press 2”). With the former, the entire prompt reads as a single descriptive phrase. Users may have interpreted the “press 2” in the latter as a separate phrase, stated as a command.

### 3.3 Summary of User Perspective

The eight design considerations above describe how system level choices will affect various classes of users. No one dialogue style will be best for all applications. We illustrated these design considerations by applying them to a few menu styles. For
example, DC2 and DC4 suggest that applications with few repeat callers may do best to use temporal menus. DC3 suggests that those applications with callers who select the same menu items on each call will do well with a style that employs absolute selection, such as standard menus. Even subtle choices such as whether to prompt for the skip key may serve certain types of users more than others, as suggested by DC8. Because of the myriad choices and tradeoffs involved, however, we encourage designers to consider how these eight points apply to their applications rather than relying on summary guidelines.

The next section further enlarges the design space for menus. The following section presents the design space for forms.

4 More Menu Styles

The first division of a menu recording identified a header, items, and a footer as the parts. Section 3 presented some of the styles that come from design choices about how to recombine these parts. This section begins with a description of some other combinations that yield plausible styles. Moreover, it is useful to subdivide each of the parts, particularly the menu items, to further expand the design space.

4.1 Combinations of Header, Items and Footer

The only explicit transition considered in section 2 was a “next item” action. The only timeout actions considered were repeat of the current item, movement to the next item, and restart of the menu after the footer. Other choices are possible. We list some and mix and match them to generate two additional menu styles.

An obvious additional explicit movement command is “previous item”. A generalized form of the “next item” action is relative numeric movement: when a listener presses 3, the computer advances by 3 items. Menus could also include absolute movement
commands that move to a fixed position regardless of the listener’s current location. For example, pressing 3 would move to the third item.

The cautious style, shown in Example 4, employs timeout advance to the next item, explicit absolute movements, and absolute selection. A user can press a number associated with an option to jump to it. If the user presses the number associated with the current option, it is selected. A user who is not so cautious can press the number twice in sequence, without waiting to confirm that it is the correct option. A variant on this style would use positional selection, so that users pressed a single selection key once positioned on the correct item, rather than pressing the number associated with it.

Welcome to the ABC Bank’s bank-by-phone. You can jump to any option by pressing the number associated with it.

Account balances. ^ press 1:  
[pres ses 4, interrupting]  
Open a new account, press 4 again.  
[pres ses 3]  
Transfer money between ^ accounts, press 3 again.  
[pres ses 3]

Example 4: A cautious menu style dialogue.

The cautious style may be especially useful to occasional users who become somewhat familiar with the contents of an application without memorizing all the options that may eventually interest them. Frequently, such users may remember that the desired option is somewhere near the end of the menu, without remembering its exact number. This style assumes that it’s easier to prevent errors than to recover from them: it makes it safe to guess the number. The style is also reasonably effective for mechanism experts who want to skip through all the options quickly. It is not quite as effective, however, as having a single skip-ahead key because the user has to press one, then two, then three, rather than pressing the same key repeatedly. Remembering the number for the next position is an added cognitive burden, as well as being mechanically more cumbersome. Finally, the
style functions similarly to standard menus for complete novices, as long as they select the current option just after the number for it is announced.

Timeouts can be used to escape from the menu or to make selections, instead of or in addition to advancing to the next item. For example, call processing applications often use a variant of standard menus where a timeout at the end of the menu escapes to a human operator, rather than repeating the menu. Some even repeat the menu once and then escape if the user still has not selected an option.

*Rejection menus* (Example 5) are one style where a timeout selects the current option. In this style, users press a button to reject the current option or wait to select it. This style is a counterpart of the temporal style described above where users press a button to accept the current option or wait to hear the next. If it is desirable to use only one button, rejection menus have some advantages over temporal menus, particularly when a user wants to select an option late in the menu and can quickly reject some of the earlier ones (see DC6 above).

Welcome to XYZBank's bank-by-phone.

If you'd like account balances, please wait. Otherwise, press #.

{presses #}

Transfers between accounts. Wait or press # to reject.

{presses #}

Mortgage rates. # to reject.

{presses #, interrupting}

Open a new account. # to reject.

{waits, causing this option to be selected}

Example 5: A sample rejection menu dialogue.

4.2 Subdivision of Menu Items

It is useful to further subdivide the header, item, and footer recordings, both to clearly identify the kinds of information they convey and to allow for explicit movements
between subparts. We illustrate this subdivision and recombination process for menu items.

A menu item can contain three kinds of information: a description of the contents, an indicator of whether it is already selected (when multiple selections are allowed) and information about available actions. The selection indicator could be a tone, a word such as "selected," or a change of voice for the option description (e.g., male instead of female). Just as the header, items, and footer can be recombined in more than one way to form different styles, so the subparts of menu items can be recombined in different ways.

Typically, timeouts move through the subparts in a particular order, giving the effect of a menu item as a single part. In that case, the order of the subparts is critical, as suggested by D.C. In styles that provide an explicit skip key for movement to the next menu item, it is generally best to put the action prompts at the end, where mechanism experts will not be bothered by them. For standard menus, too, most researchers agree that the selection prompts should follow the option prompts ("for X. press 3") [Engelbeck and Roberts 1990; Halstead-Nussloch 1989]. Thus, it appears that the action prompts should follow item prompts in all menu styles.

There is a tradeoff in whether to include action prompts for actions that are helpful but not necessary in interacting with the menu. For example, during pilot testing for the experiment cited above, we tried three variants of the standard with skips style. All three told users in the header that they could press # to skip ahead. One variant did not mention # in any of the menu items. Some of our pilot subjects pressed # to skip menu headers, but never guessed that they also could skip through the options. The second variant mentioned pound after each item. Subjects were very slow initially. The third variant told them in the header and the first item in each menu, but not thereafter. We chose this last variant for the final study because it produced the best overall performance in the pilot test.
Menus can also include explicit actions for moving among the subparts of an item. One possibility is to provide a "help" action that moves from the option description to the beginning of the prompts for available actions. Consider the fast standard style, illustrated in Example 6. Like standard menus, it uses timeouts to advance and numeric selection. Each item consists of a terse option description and a prompt to press the number associated with the option. Unlike standard menus, however, the prompt for the number is not played unless the user presses the help key (0). Thus, while listening to the menu items, the menu sounds like a temporal menu, since it includes only option descriptions, not selection prompts. For users unfamiliar with the menu contents, this allows them to hear the options more quickly than with the standard style. Once the user becomes familiar with the menu contents, however, it is still possible to type ahead a numeric selection, which would not be possible with temporal menus.

Although the fast standard style may be worth exploring further, it has one major drawback: it assumes users will know to press 0 when they want to find the number associated with a particular option. The header can include a prompt that mentions this, but that may not be enough for first-time users. If the menu items include a prompt telling users to press 0 to find out how to select, that prompt will take more time to recite than the numeric selection prompt.

Welcome to the ABC Bank's bank-by-phone. Press 0 when you hear the option that interests you.

Account balances:

Transfer money between accounts:

Mortgage rates:

Open a new account ^:
  [presses 0]
Press 4 to select this option. In the future, you can press 4 to select this option any time during the menu. ^
  [presses 4]

Example 6. The fast standard style.
4.3 Prompt Wording

Variations in wording of prompts can give two styles a very different sound and feel, even if they are identical on all the system design dimensions. For example, a variation of the standard with skips style was designed for obsessive-compulsive psychiatric patients who could not tolerate ambiguity [Sorce, et al. 1993]. It included both a short prompt for each menu item (e.g., “checking account balance”) and a longer one immediately following (e.g., “the account balance for checking account number 1042030776.”)

The yes-no style, illustrated in Example 7, is a variant of 2-button menus. The differentiating factor is that the yes-no style uses interrogative prompts phrased as yes-no questions whereas the 2-button style uses descriptive prompts. The first item in a yes-no menu uses a full sentence question and subsequent items use fragments that omit the initial interrogation phrase (“Do you want...”).

Welcome to the ABC Bank’s bank-by-phone.

Do you want to hear your account balances? 1 yes, 2 no:^
[presses 2]

transfer money between accounts?, 1 yes, 2 no:^
[presses 2]

mortgage rates? 1 yes, ^ 2 no;
[presses 2, interrupting prompt]

open a new ^ account? 1 yes, 2 no;
[presses 1, interrupting prompt]

Example 7. The yes-no style, a variant on 2-button menus.

Another variation is to have prompts for actions draw on spatial analogies. For example, in a 2-button menu style variant, the keys on the telephone keypad can be used as cursor
keys (4 left, 6 right, 2 up, 8 down) [Roberts and Engelbeck 1989; Rosson and Mellen 1985]. The action prompts could be. "Go right for the next option; left for the previous; down to select the current option; up to exit this menu without making a selection".

4.4 Lists: a Special Case of Menus

Many audio applications also include lists as dialogue components. For example, voice mail applications allow a user to move through a list of messages in a mailbox. We can view lists as degenerate cases of menus that allow movement but not selection. For example, a conventional cassette-based answering machine provides the analog of temporal menus: it plays the messages one after the other, using timeouts to advance between messages. All the menu style variations (except those relating to selection) are equally applicable to list styles.

One list style is worth analyzing because it includes several unusual movement commands. We call it the radio-scanner style because it makes an explicit analogy to the radio scanners found in many automobiles [Kondziela 1990]. The radio scans from station to station, playing a few seconds of each until the user presses a button to stop scanning. Similarly, the radio-scanner style advances from item to item via timeouts, playing just a headline of each item. In addition to timeout movements, # and * are explicit move forward and move back commands. ## moves ahead by five and ** moves back by five items. This restricted use of relative numeric movement still leaves all of the numeric keys available for absolute movement. Rather than pre-assigning numbers to the items in the list, each user can assign numbers to favorite items. The user does so by pressing the numbered button twice while listening to that item. Thereafter, pressing that number once initiates an absolute movement to that item.

The radio scanner list style also includes an explicit action for movement within an item. Each item consists of a headline separate from the rest of the item. To get from the
headline to the rest of the item, a user presses a button (0 in this case). This is quite similar to the explicit help variation used in the fast standard menu style, but here when a user presses 0 the computer plays the rest of the item.

4.4 Summary of Audio Menus

The primary design space for menus comes from a division of the recording into a header, items, and footer, and the division of the select action into marking and termination. We presented several styles in this space and suggested applications to which they might be especially well suited. We expanded that design space through a recursive application of the subdivision and recombination framework to the menu items. Even within a single menu item there were opportunities for explicit transitions, as the fast standard style illustrated. Finally, variations in prompt wording can give two implementations of the same system design choices a very different sound and feel. We turn now to audio forms.

5 Forms

Forms guide people through the process of entering several related pieces of information. This section begins with three sample form styles, to illustrate some of the potential diversity. Then, the analytic framework of dialogue mechanism decomposition and recombination generates a design space for form styles. In this case, the recording parts are a header, a footer, one entry blank for each piece of information to be entered, and optionally a review node associated with each entry blank. The actions insert and remove values from entry blanks. We then recursively subdivide the entry blanks: two additional form styles illustrate points in the expanded design space.

5.1 Three Form Styles

The first and most easily understood telephone form was part of the PhoneSlave [Schmandt and Arons 1984], which took phone messages when its 'master' was away from his desk. It used a conversational style. The system asked each caller a series of
questions ("Who's calling please", "What is this in reference to?", "At what number can he reach you?", etc.) After playing a question, it recorded whatever the caller said, until a long pause was detected, then went on to the next question. Example 8 illustrates this style for a classified advertising application where the user enters information about a car for sale.

What kind of car are you selling?
["Cadillac"]

Please enter the year? For example, enter eight-six for a 1986 model.
[presses 9, then 1]

What color is the car?
[says "Gray... well, more bluish-gray"]

Enter your phone number.
[presses 2-2-2-9-9-9-9]

---

Example 8. The conversational form style.

One drawback of the conversational style is that a user cannot correct mistakes. The careful style (Example 9) resembles the conversational style, but automatically reviews each entry [Sorce, et al. 1993]. If the user confirms the value, the form continues with the next entry blank. If the user cancels it, the form prompts the user to enter a different value. This style was used by obsessive-compulsive psychiatric patients. They filled out the same form once each week. Each entry blank contained a multiple choice question, presented through a standard with skips style menu, described above in section 2.
What kind of car are you selling?
[“Cadillac”]
You said, “Cadillac” (plays back recording). If that’s right, press #. If not, press *.
[Presses #]
Please enter the year? For example, enter eight-six for a 1986 model.
[Presses 9, then 1]
A 1991 model. If that’s right, press #. If not, press *.
[Presses #]
What color is the car?
[says “Gray... well, more bluish-gray”]
You said, “Gray... well, more bluish-gray”. If that’s ^
[Presses *, interrupting prompt]
What color is the car?
[says, “Bluish gray”]
You said, “Bluish gray.” If that’s ^
[Presses #, interrupting prompt]
Enter your phone number.
[Presses 2-2-2-9-9-9-9] 222-9999. If that’s right, ^
[Presses #, interrupting prompt]

Example 9. The careful form style.

The user-controlled style (Example 10) gives users even more control, both over initiation of value entry and over review of values. Users can gather their thoughts before starting to record, and can skip entry of values they consider irrelevant. After entering a value, the form continues automatically with the next entry blank, but the user can choose to go back to an entry blank, review the value there, and replace it.
Brand. To record, press 1. End recording by pressing #. For the next entry blank, press 9.

[Presses 1, says "Cadillac", presses #]

Model year. To enter a value, press 1. To review the previous entry blank, press 7. For the next entry blank, press 9.

[Presses 1]
Enter two digits. For example, enter eight-six for a 1986 model.

[Presses 9, then 1]

Color. To begin recording, press 1.

[Presses 1, says "Gray, well, more bluish-gray"]

Phone number. To enter a value, press 1.

[Presses 7]

Color. "Gray, well, more bluish-gray". To replace this recording, press 1.

[Presses 1, says, "Bluish-gray"]

Phone number. To enter a value, press 1.

[presses 1]
Enter your seven-digit phone number at any time.

[presses 2-2-2-9-9-9-9]

That’s the end of the form. If you’re satisfied with this ad and would like to save it, press 3.

[presses 3]

Example 10. The user-controlled form style.

These three styles only hint at a larger design space. For some applications and user populations, conversational forms may be appropriate. When the consequences of incorrect entry are high, however, some method of allowing review should be provided, either automatically or upon user request. If some of the entry blanks are optional, or user initiation is desirable, the user controlled style or variations on it may be the most suitable choice. Sometimes, none of these three styles will be quite right. For example, if the consequences of incorrect entries are very high, it may be appropriate to have explicit initiation of value entry, as in the user-controlled style, but still automatically review each value, as in the careful style.
5.2 Design Dimensions

We can gain more insight into the style variations by exploring the design dimensions associated with recombining dialogue parts. The recordings in a form divide naturally into a header, a footer, entry blanks, and review nodes associated with entry blanks. For example, the first entry blank in Example 9 (the careful style) stated, “What kind of car are you selling?” After the user recorded, “Cadillac”, the review node stated, “You said, ‘Cadillac’. If that’s right, press #. If not, press *.” To conserve space, the examples omitted the headers and, where possible, the footers; they function analogously to their counterparts in menus. The primitive actions in a form are insertion and deletion of values from entry blanks. The design choices include which actions are available for movement, how value changing actions are bundled, and whether they are initiated explicitly or with timeouts.

5.2.1 Movement Actions

The possible transitions for movement among the entry blanks are analogous to those for movement among items in a menu. Since users will typically enter information in most or all of the entry blanks, movement forward and back by one entry blank are appropriate movement commands. From the review nodes, users can return to the associated entry blank or transition to the next one. The backward movement action is one factor that influences users’ ability to review, especially if there are no review nodes. In the user-controlled style, it allows review of values at the user’s discretion, by explicitly moving backwards.

All the form styles in this paper include a subset of these movement actions. Some applications, however, may benefit from additional mechanisms. For example, to accommodate selective review of long forms, absolute or relative numeric jumps might be included, or actions to move to the next empty or the next already filled entry blank.
An action to jump to the footer may be useful after reviewing the contents of a few entry blanks. The inclusion of more explicit transitions does not necessarily increase the complexity for users. If, however, the system includes prompts for the additional transitions, mechanism novices will pay a time penalty (DC7).

5.2.2 Action Combinations

All form styles need some way to add a new value to an entry blank. Some styles may include a delete action as well, to remove a value from an entry blank.

Several composite value change operations may be included. For example, a delete-all command erases not just one but all the values in an entry blank. A replace action is a delete followed by an insert. A replace all command would delete all the current values and then initiate insertion of a new one. As described in DC1, there is a tradeoff between flexibility when actions are separated and simplicity when actions are combined.

The insertion action is often bundled with a transition to the review node or the next entry blank. After initiation of value entry the form passes control to a subroutine. The subroutine may allow the user to record, enter a sequence of touch-tones (e.g., a date) or select from a menu. When the value entry subroutine returns, the form follows the bundled transition. The conversational and user controlled styles transition to the next entry blank while the careful style transitions to a review node. From the review node, a user can erase the value and return to the current entry blank, or go on to the next entry blank.

Of course, it is not necessary to bundle the insertion action with a movement in this way. After returning from the value entry subroutine the system could replay the contents of the current entry blank. This may be appropriate either as an alternative mechanism for reviewing the contents of the current entry blank, or to encourage the addition of several values to the entry blank.
Other composite actions may also be included. For example, an undo command first moves to the previous entry blank, and then deletes a value there. Similarly, from a review node, the undo command would return to the current entry blank and erase the value just entered.

### 5.2.3 Positional vs. Absolute Initiation

It is theoretically possible for an absolute mechanism to initiate value entry or removal: from anywhere in the form, a user could initiate addition of a value to any of the entry blanks. All the form styles in this paper, however, utilize positional initiation: insertion and deletion actions apply to the current entry blank.

### 5.2.4 Effect of timeouts

Entry of a new value can be initiated with an explicit action, or with a timeout. This choice may have the single largest effect on the feel of the form. Explicit initiation of value entry gives users control over the pace of the interaction, allowing them to gather their thoughts before entering information. On the other hand, timeout initiation can make the dialogue flow naturally for novice users (DC4).

When value insertion is the only action available from an entry blank, it can be initiated by default, even if the user does not wait for the timeout at the end of the entry blank. For example, in an entry blank that expects input of data by touch-tones, the careful style interprets any user input as a date.

From a review node, timeouts can initiate any of the possible actions. For example, in the careful style, callers must either explicitly erase the value and return to the current entry blank, or confirm it and go on to the next entry blank. If the user does neither, the system repeats the prompt. An alternative version would treat silence as assent (timeout moves
ahead to the next entry blank) or dissent (timeout erases the value and returns to the current entry blank).

5.3 Entry Blank Subdivision

As in menus, it is useful to apply the subdivision idea recursively, in this case to entry blanks. This highlights the types of information that an entry blank normally includes and opens additional design choices for recombination of the parts. In particular, we consider making the insertion and deletion commands sensitive to the current position within the entry blank.

Three kinds of information can appear in an entry blank: a description of the desired values, current values (e.g., recordings), and prompts for actions. The descriptions of desired values can either be descriptive (e.g., “The color of your car”) or commanding (“Record the color of your car”). DC8 suggests that the choice of timeouts or explicit initiation of actions interacts with the choice of wording styles for these descriptions. In informal tests, we found command statements to be less effective with explicit initiation of value entry. When entering dates, users often forgot to press 1 to initiate data entry. They proceeded to enter several touch-tones (e.g., 0-7-3-1 for July 31), which the system interpreted as explicit user actions rather than as entry of a date.

Thus far, the discussion has assumed that value change commands such as insert and delete have the same effect from anywhere in an entry blank. When entry blanks can include multiple values, position sensitive actions can be helpful. Consider an entry blank that has several dates, each entered by touch-tones. An absolute deletion command would always remove the last date. A positional deletion action could remove the date currently being played. Likewise, the user could insert an additional date just before the current one.
The same idea applies to entry blanks that contain recorded voice. Positional insertion and deletion are especially useful in dictation applications. With positional insertion, when a user inserts a new recording the computer splits the recording then playing into two segments and inserts the new recording in the middle. Similarly, a positional deletion action could remove the voice segment currently playing back. An even more complex positional deletion action would require the user to mark the beginning and end of the voice portion to delete. Even in fairly complex dictation applications, however, the simpler mechanism of deleting the entire voice segment currently playing might work quite well.

Two variants of the user-controlled style illustrate the design possibilities for entry blanks. The event calendar style [Resnick 1993] allows the general public to add new event announcements to a public bulletin board. Callers fill out forms with entry blanks for a headline, date, time, location, sponsor, contact phone number, and details. The style has evolved over the more than two and a half years that the application has been used by the general public.

The event calendar allows multiple values. This is frequently useful for dates and for appending additional thoughts in the details entry blank. In Example 11, the user, who is filling in a form for a classified ad, enters two telephone numbers. The delete action deletes all the voice in the entry blank, not just the last segment recorded, because informal tests indicated that was too confusing for novices. The delete action does, however, delete only the last date if more than one has been entered in an entry blank.

---

Brand. The kind of car you are selling. You can gather your thoughts before starting to record. When you're ready, press 1. End recording by pressing #. For the next entry blank, press 9.

[Presses 1, says “Cadillac”, presses #]

Year. The model year of the car. To enter a value, press 1. To review the previous entry blank, press 7. For the next entry blank, press 9.

[Presses 1]

Enter two digits. For example, enter eight-six for a 1986 model.
Color. The color of your car. To begin recording, press 1. ▲
[Presses 1, says “Gray, well, more bluish-gray”]
Phone number. The number people should call if they want to buy your car. To enter a value, press 1. To review the previous entry blank, press 7. ▲
[Presses 7]
Color. “Gray, well, more bluish-gray”. To replace this recording, press 1. ▲
[Presses 1, says, “Bluish-gray”]
Phone number. The number people should call if they want to buy your car. To enter a value, press 1. ▲
[Presses 1]
Enter your seven-digit phone number at any time.
[presses 2-2-9-9-9-9-9]
That’s the end of the form. ▲
[presses 7]
Phone number. 222-9999. To enter an additional value, press 1. ▲
[press 1]
Enter your seven-digit phone number at any time.
[presses 3-3-8-8-8-8-8]

Example 11. A sample dialogue with the event calendar style. Note that in this style, entry blanks initially have long descriptions. They are omitted once a user has entered a value.

The dictation style is another variant of the user-controlled style. It permits multiple values in entry blanks and employs positional insertion and deletion actions. The style might be well suited to the dictation of entries for a patient chart in a hospital. The patient chart provides a single place where all doctors and nurses who minister to a patient make notes intended for the use of the other caretakers (and for the courts in the case of malpractice claims). In most hospitals, patient chart entries are free-form, either written by hand or dictated and then transcribed. There is no pre-defined structure, but each hospital has conventional genres for what information to include in what order for specific kinds of entries. Some of these conventions could be embedded in forms. Such forms would have symbolic fields, such as the doctor’s id and drug prescription quantities, as well as unstructured fields to accommodate spoken notes. Doctors and nurses could add user-defined structure by recording several separate segments in each entry blank.
Brand. The kind of car you are selling. Record, 1; Next entry blank, 9.
[Presses 1, says “Cadillac”, presses #]
Brand. “Cadillac”. Record, 1; Erase, 2; previous entry blank, 7; next entry blank, 9.
[Presses 9]
Year. The last two digits of the model year of the car. Enter value, 1; Next entry ^...,
[Presses 1, interrupting prompt]
Enter two digits. For example, enter eight-six for a 1986 model.
[Presses 9, then 1]
Year. 1991. Enter value, ^...,
[Presses 9, interrupting prompt]
Color. The color of your car. ^
[Presses 1, says “Gray, well, more bluish-gray]
Color. “Gray, well, ^ more bluish-gray”...,
[Presses 1, interrupting, and says “Actually, it’s”, then presses #]
Color. “Gray, well, actually, it’s more bluish-gray” ^
[Presses 9]
...and the dialogue continues.

Example 12. The dictation form style.

5.4 Form Styles Summary

Form design vectors summarize the design choices about movement and value change actions and how they are combined. Figure 8 shows the design vectors for the five styles described. The first three show the effect of design choices about movements between entry blanks and actions that change values in entry blanks. The last two illustrate design choices for actions within individual entry blanks. For the sake of simplicity, the design vectors omit information about which subparts to include in each main part and how to word prompts.
<table>
<thead>
<tr>
<th>Action</th>
<th>conversational</th>
<th>careful</th>
<th>user controlled</th>
<th>event calendar</th>
<th>dictation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next entry blank</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Previous entry blank</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value Change</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add value</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Timeout Initiation</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explicit Initiation</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go to review node?</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go to next entry blank</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Delete value</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Multiple values?</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Positional insertion and deletion?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

Figure 8. The form design vectors for four styles. Some of the movement actions that are not used in any of the styles are omitted for the sake of brevity.

There have been no controlled comparisons of form styles in the literature, and we do not report any in this article, but it seems likely that the appropriate style choice depends on the tasks and levels of user experience. The styles that include more actions, simple rather than composite actions, and make less use of timeouts afford greater user control. Such styles also demand greater user control, however, which may be difficult for novice users.

6 Conclusion

Menus and forms are important dialogue structures in telephone-based interactive voice response applications. Lists, a third important structure, can be viewed as degenerate versions of menus: they allow movement between items but not selection. The commercial marketplace and much of the research literature are currently dominated by one menu style and one form style. While these styles are fairly easy for novices to learn, they are limiting because the locus of control rests with the machine rather than the user.

Fortunately, many other interaction styles are possible. This article has presented twelve menu styles and five form styles, including all of the interesting styles reported in the literature. The design spaces used to describe these alternatives can also be used to
generate new styles. We had not considered all of the styles presented in this paper until we had constructed the dimensions of the design space.

With so many styles to choose from, and the possibility of generating custom styles, the designer can tailor the choice to the needs of particular applications, just as screen-based interface designers choose menu styles (e.g., pull down, pop up, radio buttons) to match particular needs. For example, if users will frequently browse among large collections of options, it will be worth their initial effort to learn the commands for movement among options in menus. If some frequent users will memorize the contents of menus, numeric selection may be best. If some entry blanks in a form are optional or if users will need time to gather their thoughts before recording, a form style with explicit commands to initiate value entry may be appropriate. Rather than relying on these summary guidelines, however, we encourage designers to describe their expected users and apply the design considerations of section 3 to a range of possible styles.

While the design spaces are useful to researchers and designers of telephone-based dialogues, they have much broader implications. Audio interactions with workstations and personal digital assistants will be useful whenever the users' eyes are busy with other tasks. Moreover, much of the analysis in this article carries over to any interface that is temporal in nature, even if it is not audio. In general, if only a small part of the relevant information can be presented at one time, then all of the fundamental audio design choices will be relevant: what information to include in menu options and form entry blanks, how users can navigate among the pieces of information, and which actions to initiate with timeouts versus explicit commands.

Consider, for example, the new generation of ADSI-compatible display phones [Bellcore 1992]. These phones have displays that are 20 characters wide and up to eight lines long. There is a signaling protocol that allows these phones to receive 160 character batches during the course of normal voice calls. One natural idea is to augment interactive voice
response services with a presentation of menus on the screen. If one option is displayed on each line, it may be possible to display the entire menu at once, but 20 characters per menu option leads to cryptic descriptions of the options. Another possibility would be to display one option at a time, using the full screen space to give a clearer description of the option. Since only one option at a time will be displayed, many of the audio menu style variables apply, including how users move through the options and make selections.

Similar design considerations apply to other small-screen devices, such as the screen employed in in-flight telephony services [Kar is, et al. 1993] or a palmtop virtual reality system [ Fitzmaurice, et al. 1993]. These authors point out the difficulty in presenting the necessary information in a limited screen space. Analogs of the menu and form styles presented in this paper may apply to such devices.

The temporal presentation of audio creates interesting design challenges. Exploration of techniques that give users control over the time dimension are just beginning. The essence of that exploration is to break information chunks into ever smaller parts and find natural ways for users to control which part will be presented next. The design spaces in this article have applied that principle to both menus and forms. The division of menus and forms into component parts, and those parts into even smaller parts, opens up new possibilities. It may be that the best styles have not been invented yet.
References


