Palo Alto Research Center

Voice Annotation and Editing in a Workstation Environment

Stephen Ades and Daniel C. Swinehart

XEROX

Voice Annotation and Editing in a Workstation Environment

Stephen Ades* and Daniel C. Swinehart

CSL-86-3 September 1986 P86-00058

A version of this paper appeared in *Proceedings of the AVIOS '86 Voice Input/Ouput Systems Applications Conference*, Alexandria, VA, September 16-18, 1986; American Voice Input/Ouput Society, Palo Alo, CA 13-28.

CR Categories and Subject Descriptors: D.2.6 Programming Environments **[Software Engineering]**; E.5 **[Files]**; H.1.2 User/Machine Systems **[Models and Principles]**; H.4.1 Office Automation **[Information Systems Applications]**; H.4.3 Communications Applications; I.7.1 Text Editing **[Text Processing]**; I.7.2 Document Preparation.

Additional Keywords and Phrases: Design, human factors, voice annotation, voice editing, multi-media documents, workstation applications.

*Author's present address: Stephen Ades, University of Cambridge Computer Laboratory, Corn Exchange Street, Cambridge, England, CB2 3QG

Xerox Corporation
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304

VOICE ANNOTATION AND EDITING IN A WORKSTATION ENVIRONMENT

Abstract: The Xerox Cedar experimental development environment, running on personal workstations, incorporates a structured document editor, which is used as the basis of many applications, including programming and document preparation. A project at the Xerox Palo Alto Research Center has integrated the telephone into this environment, partly to afford control over a user's telephone from his workstation and partly to incorporate recording and playback of voice into the workstation capability.

This paper describes incorporation of voice annotations into normal documents within this environment. The user interface is designed to be lightweight and easy to use, since spontaneity in adding vocal annotations is essential. Voice within a document is shown as a distinctive shape superimposed around a character, so that the document's visual layout and its contents as observed by other programs (e.g., compilers) are unaffected. Users point at text selections and use menus to add and listen to voice.

Simple voice editing is available: users can select a voice annotation and open a window showing its sound profile. Sounds from the same or other voice windows can be cut and pasted together, and a lightweight "dictation facility" that uses a record/stop/backup model can be used to record and incorporate new sounds conveniently. Editing is done largely at the phrase level (never at the phoneme level), representing the granularity at which editing can be done fastest and with least effort. The voice itself can be annotated with text. This and several other features have been designed to add speed and meaning to the editing process. The dictation facility can also be used when placing annotations straight into documents.

This paper describes the user interface in detail and explains why we believe that this unusually lightweight interface is the desired abstraction for voice in a general purpose workstation. It also discusses the supporting system capabilities provided by a distributed environment on a local area network.

1. Introduction

There has been a trend in office systems for some time toward a single computer-based system that can satisfy all of a user's working needs. Combined with a trend away from large time-shared systems and toward personal computers, there has evolved the concept of the *workstation*, a powerful personal computer that performs a wide range of functions. These may include text composition, document preparation and typesetting, support for program development, spreadsheets and accounting facilities, and general-purpose local computing, as well as access to remote computers. Today's workstation is usually equipped with a high-resolution display, a keyboard, a mouse or other pointing device, a network connection, and possibly a high-capacity local disk. The workstation environment of the near future will also include scanning equipment and document architectures that can describe scanned images [4], thereby taking over the role of the facsimile machine.

The office telephone and the workstation together satisfy nearly all of a user's communications needs. It is therefore useful to consider combining them into one system. This affords two major possibilities:

- Modern office telephone systems offer all kinds of facilities, such as call forwarding, dialing by name or abbreviation, and three-way conferencing. These are invariably cumbersome to use, hard to learn, and hard to remember, since the user interface is limited to a twelve digit keypad, some progress tones, and perhaps a few special-purpose buttons and lights. We believe that the user would prefer to perform these functions using the power and convenience of such workstation facilities as on-screen menus, text editors, and comprehensive informational displays. The user should be able to master more of the capabilities of the telephone system and to accomplish telephone-related tasks more quickly and easily.
- We are beginning to see document architectures integrating text, graphics, scanned images, and other visual media [3, 4, 18]. Integration of voice (or rather of sound in general) into documents promises a wide range of new applications, from voice mail to documents containing simple vocal annotations and even to documents whose self-contained "scripts" can generate automatic audio-visual presentations.

Work on the integration of the telephone and the workstation (the *Etherphone* project), including applications of digitally-recorded sound, has been in progress at the Xerox Palo Alto Research Center for some time. An overview of the goals and architecture of this project has been reported previously [16].

To understand the aims of the Etherphone project, it is worth distinguishing them from those of other activities to integrate voice and data, notably the *Integrated Services Digital Network*, or *ISDN*. ISDN is a set of standards emerging from the telephone industry that will permit the use of existing telephone networks and other communications networks to carry all kinds of traffic, including voice, data, and eventually video, in digital form. The ISDN is concerned only with the integration of these media for the purpose of transmission and switching, not with their integration in user applications. In contrast, the Etherphone project attempts to answer the following question: Independent of the

methods used to transmit and manage the information, can we extend our workstation environment to include telephony and recorded voice, so that the powerful interactive techniques that have been developed to deal with more conventional computer-based information and applications can be applied to this new medium? We are also concerned with the production of a complete architecture for voice that encompasses recorded voice applications and office telephone management. Applications programmers can use the components of this architecture to construct a range of innovative voice capabilities.

This paper describes recent work on the integration of voice into a document architecture, based on the facilities of the Etherphone system. After a review of the capabilities of existing systems for integrating voice and other media, we discuss the requirements of such an architecture and of the user interface, placing particular emphasis on ease of use. Our editing interface is described in some detail. Finally, we discuss the systems components needed to support our voice applications.

2. Conflicting Requirements of a Voice Interface

A number of systems combining voice and visual media have been built. In trials using one of these systems [10], it was found that people will not use voice annotation unless it is made available as a simple and fluent process, encouraging spontaneous and off-the-cuff comments. Many of the systems we will be mentioning are products that, recognizing this need, include a set of fast and convenient facilities for voice annotation.

As an illustrative example, one such product, the Centerpoint System [2], supports the executive-and-secretary paradigm. The secretary is provided with a keyboard and touch-sensitive screen. The executive, who is presumed to have neither the time nor the skill to use a keyboard, dictates letters, reports, and memos for the secretary to type. Subsequently, the executive can mark the typed copy with further vocal annotations that the secretary later uses to correct the document.

Consider some other professions that could benefit from a simple voice annotation system. Members of an engineering group could send drafts of reports or documentation to other group members, who would then annotate them with voice messages and return them. Teachers could even use annotation for grading assignments. A lecturer would receive a completed assignment by electronic mail, add comments in spoken form, and then return the whole thing to the student.

On its own, a simple annotation interface cannot provide a number of important capabilities, including the ability to edit the annotations themselves, and to provide these facilities as part of a general architecture that can be extended to include future applications.[1]

Recently, some general-purpose systems have begun to emerge. Built by computer science research teams, they are intended to be flexible, extensible, and to provide unified user interfaces linking a variety of different media. The most notable effort to date is a collaboration of several

[1] In fairness, most of the existing systems that support annotation, although designed for specific purposes and strictly limited in the capabilities they provide, do include some voice editing facilities.

research organizations built around a common multi-media document architecture [5, 11, 18]. In these systems, graphical objects typically are placed in text by opening a graphics "subwindow" within the text. Further text may be placed inside this graphics subwindow by including a text subwindow within it, and so on; the structure is general and flexible. Such systems allow voice to be inserted in a equally general manner. For example, the user opens a voice subwindow (consisting of a voice "icon" and some identifier for the speech), then talks into a microphone. If the user wishes to edit the resulting annotation, some kind of energy profile can be made to appear in a separate window, as a plot of the speech waveform against time. This profile provides a representation that can be manipulated by manual editing operations that are interpreted as edits to the actual voice. These editing capabilities tend to support exacting, detailed modifications to the voice, rather than the simple operations that are needed for annotation.

The point of these examples, taken from real systems, is that there is a conflict between user interfaces that are convenient for simple dictation and document annotation, and those that provide general-purpose manipulation of voice and other media.[2]

We believe that a proper trade-off between the desire for generality and the need for a user interface with the right degree of spontaneity is a hard one to achieve. In the following description of a voice editor, we hope to show that an acceptably wide range of voice editing applications can be presented through a user interface that permits both simple annotation and more elaborate manipulation of recorded voice. While the goal was to retain flexibility and generality, careful thought has been given to the user's model of what the interface does. If an operation could not be expressed in terms of a reasonably simple user-level model, or if the result of some manipulation might not be clear to the user under all circumstances, then it was not included in the interface, regardless of its utility in some contexts.

3. An Approach to Voice Integration

This section outlines the methods we have chosen for addressing the trade-offs between ease-of-use demands and the requirements for powerful and flexible voice editing capabilities. We discuss the power and the limitations of a uniform approach to editing all visual objects. We then outline the approaches to voice annotation and to voice editing that we have used in the Cedar voice editor.

3.1 Strengths and weaknesses of a uniform editing interface

An essential feature of a good integrated editor is uniformity. Uniformity means that the user learns a consistent set of methods for selecting a variety of visual objects, and a consistent set of commands, each of which will have a similar effect on the selected objects. This common interface

[2] This conflict is not limited to the voice domain. That is, the same sort of conflict between providing simple methods for common operations and allowing maximum flexibility and power for more elaborate requirements arises in text editors, illustrators, and many other interactive applications.

hides from the user any implementation differences required to represent the behavior of the different visual media.

One example—the editor supplied with the Xerox 8010 (Star) workstation [15]—will suffice to illustrate this concept. In Star, the same sequence of keystrokes and mouse-button clicks can be used to move a portion of text from one textual window to another, to move a geometric figure or a digitized photograph from one location to another within a graphics window, or to move an entire document from one "file folder" to another. Deletion of any of these objects can be accomplished by moving the selected object to an iconic representation of a wastebasket.

Our basic approach, like that of most of the voice editing systems we have studied, is to extend the uniform editing concepts of an editor for visual media to encompass voice as well. But we believe it is equally important not to stretch this principle to its logical limit. Instead, we need to determine those concepts that cannot beneficially be applied uniformly to voice, and to add some specialized operations dealing with characteristics of voice that are unparalleled in other media. We believe the result has some important advantages over earlier integrated voice editors.

3.2 An approach to voice annotation

To insert a graphical illustration into a textual document, a common approach using an integrated editor is to create a graphics subwindow within the text, then to create and modify the illustration directly within that subwindow. Textual annotation of the illustration may be added by creating a further text subwindow within the graphics subwindow; and so on. This model is perfectly natural and understandable to the user, and is reasonably straightforward to implement.

We feel that it is a mistake to extend this model to voice. Creating a window is a heavyweight operation which goes against the "spontaneity of use" requirement for vocal annotations. No graphical representation can convey the actual content of a voice annotation; only playback of the voice itself will accomplish that.

We prefer to indicate the presence of voice by decorating the region of text to which the annotation pertains with a simple pictorial marker. This will not alter the structure of the visual media to which it has been appended. If the editor is being used for typesetting, a reviewer might comment vocally that "this paragraph looks too long and thin". The comment refers to the form rather than the content of its target. Such a sentence becomes meaningless if its inclusion in the text causes a voice subwindow to appear and distort the layout of the text to which it is attached.

3.3 An approach to voice editing

Straightforward annotations to text can be made simply by recording single utterances from a microphone and marking their presence pictorially. Playback of complete annotations is equally straightforward. The time will come, however, when a user wishes to extend or modify a voice annotation, or when the amount of voice to be included exceeds what can be dictated all at once. What is needed is a means for editing voice.

Nearly all of the systems that support voice annotations also include voice editors. Most of these voice editors function by presenting the user with some form of visual representation of the voice and with operations on this representation that allow manipulation of the voice. Taking advantage of the static, spatial representation of what is intrinsically a temporal phenomenon, the user can access any part of an utterance, designate ranges, specify reorderings, etc., much more conveniently than would ever be possible using conventional telephone-based voice editing facilities. We have adopted this eminently sensible approach as well.

Having chosen the general approach, we gave careful consideration to the type of editing interface that would best support voice editing. When editing text, one often deletes and inserts material character by character – for example, to correct spelling mistakes. If the uniform interface model is stretched too far here, one is tempted to conclude by analogy that voice ought therefore to be edited at the word level, the phoneme level, or at even finer granularities. This we believe to be a mistake, because the way in which voice and text can most easily be generated are very different. When looking at written text, one can easily correct it character by character, whereas the idea of replacing a complete sentence because of one incorrect character is both absurd and tedious. On the other hand, most of us are adept at speaking entire words or sentences, but would find it difficult to add a single word to a spoken sentence without losing the natural flow. At even a lower level, replacing one vowel sound with another is practically guaranteed to produce unsatisfactory results [1]. Besides, user interfaces that operate at this fine level of detail are slow and tedious to use. We have concluded that a voice editor should provide operations supporting the modification of voice documents at the phrase (or sentence) level, using naturally-occurring pauses in the speech to denote phrase boundaries.

Before turning to a description of our voice editor, we mention one more philosophical point that underlies our design. Even the best possible graphical representation of a recorded utterance will fail to convey much of its content. Therefore, it is easy to lose your place when dealing with such representations. A good editor should supply numerous methods for marking significant places in the representation. Simple temporary markers are useful for keeping track of important boundaries during editing operations. Permanent markers can be used to find significant locations within extended transcriptions.

We have developed one particular form of marking that we believe to be unique to this work. It helps one distinguish between "stale" and "fresh" voice. "Stale" voice is voice that is already in a document that one has received and is editing. When one is speaking into a microphone, that voice is "fresh." The difference lies in one's awareness of what has just been said; it is easy to produce a new version if a sentence is fresh in one's mind. The user is therefore more likely to want to modify the fresh voice than the stale: to correct the choice of words or phrasing, eliminate coughs or other distractions, eliminate unwanted hesitations in the wrong place, and so on. On a conventional dictation machine, these corrections would be accomplished through a "rewind, replay, rub out, resume recording" sequence. We believe an editing interface ought to supply operations that are at least as fast and convenient for reviewing, replacing, and extending the fresh phrases in a voice representation.

3.4 Characteristics of existing systems

In the design of this system, we looked carefully at a number of special-purpose or general-purpose systems for annotating voice. Among them are several product systems: the Centerpoint system mentioned above [2], the Wang Audio Workstation [19], and the Sydis VoiceStation [9]. We also studied systems produced as research projects: two related voice editors produced at AT&T Bell Laboratories [6, 20], the Intelligent Ear [13] and PhoneSlave [14] projects from the Architecture Machine Group at MIT, a Masters project by Mirrer [8], BBN's Diamond system [18], and a voice and telephony project at IBM San Jose [12].

All of these systems except the PhoneSlave and the IBM system provide voice editing facilities. The Centerpoint system, the Intelligent Ear, and Mirrer's voice editor present a graphical energy profile, while the others use simpler representations similar to the ones we have chosen for this work. The AT&T systems, the Intelligent Ear and Mirrer's system encourage editing at phrase boundaries, or rather at significant silent periods. The AT&T, Wang, and Sydis systems permit some form of textual annotation of voice as visual markers; the Intelligent Ear also makes novel use of speech recognition for marking the voice, as we will indicate below.

The Centerpoint, Sydis, BBN, and IBM systems provide for voice annotation of text documents. Of these, the Sydis and IBM systems use a form of annotation that does not affect the formatting of the documents, except for a notation in the margins. The PhoneSlave is a special-purpose system exploring the integration of interactive answering machines and electronic mail in a number of truly novel ways. As such, its characteristics do not fall neatly into either the annotation or editing categories. The message browsing facilities of this system do include a visual representation indicating the duration of each message segment.

The BBN system is based on a general multi-media document format. It is also capable of communicating with other systems using the interchange format mentioned in Section 2 [11]. Finally, the ISO standards body is defining a multi-media format, the ODA document architecture [4], which will eventually include voice.

4. The Cedar Editing Environment

The voice editor that we are about to discuss in detail was built using the Cedar programming environment [17], which has been developed as a research prototype by the Computer Science Laboratory (CSL) at the Xerox Palo Alto Research Center. A few notes on the editing capabilities in Cedar may therefore be useful to establish context.

Tioga is the standard text-editing program in Cedar. Tioga is essentially a high-quality galley editor, supporting the creation of text documents using a variety of type faces, type styles, and paragraph formats. Tioga is unusual among text editors in that its documents are tree-structured rather than being plain running text. This makes possible such operations as displaying only the section headings or key paragraphs of a document; this means that scanning a Tioga document for a particular

section can be done quickly and effortlessly. Finally, Tioga includes the ability to incorporate illustrations and scanned images into its documents. Tioga can create both black-and-white and full-color documents.

With such a general structure, one might expect manipulation of a Tioga document to be cumbersome. On the contrary, the editor interface has been very carefully constructed – using menus of commands, keystrokes, and a three button mouse – to afford very fast editing. In particular:

- Care has been taken to determine which editing operations are most frequently required and to make those operations very fast.
- The use of pairs of selections, specified with the mouse, makes operations such as copying and amending text very fast.
- The command interface is enriched by *accelerators*, simple key combinations that perform common sequences of operations.

In summary, the Tioga interface has a general and versatile structure, and a user interface that was designed for fast and fluent operation[3]. We have tried to follow that pattern in the voice editor.

Cedar has been designed so that other applications can employ the capabilities of the Tioga editor. These include the electronic mail system, the system command interpreter, and any tools that require the entry and manipulation of text by the user. This gives considerable unity to the editing interface, since for all the different types of application in which Tioga is used, identical keystrokes will perform identical functions. Wherever Tioga is used, all of its formatting and multi-media facilities are available. Thus, by adding voice annotation to Tioga, we have made it available to a variety of Cedar applications.

5. The Voice Editing Interface

In this section, the capabilities of the Cedar voice editor prototype are described in substantial detail. We first describe simple methods for adding voice annotations to standard Tioga text and for listening to such annotations in their entirety. We follow this with a description of more general-purpose techniques for editing voice annotations, all based upon manipulations of visual representations of the recorded voice.

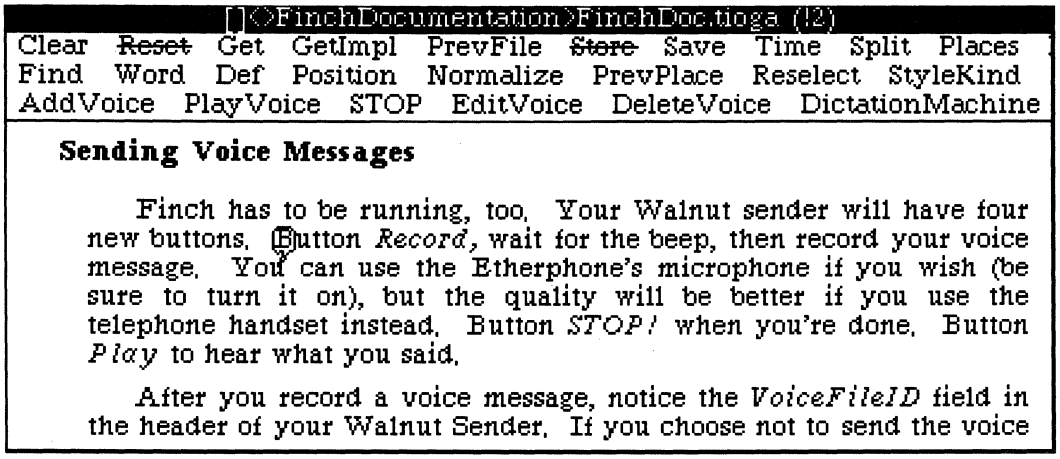
5.1 Basic annotation

Figure 1 shows a text document window, or *viewer*, from a Cedar workstation screen. Its caption defines the various regions of the viewer, indicating how one selects objects of interest and how one performs various operations on those objects. We will use the terminology defined in Figure 1 throughout the discussion.

Any single text character within a Tioga document can be annotated with an audio recording of arbitrary length. To add an annotation, the user simply selects the desired character within a text

[3] Compare with the very similar philosophy for the design of the ZOG user interface [7].

Figure 1. A typical Cedar *viewer*, containing a formatted document with a voice annotation. The top region of the viewer is a system region that includes an identifying label and three rows of *menu buttons*. A portion of the document to be edited occupies the remainder of the viewer. The user interacts with the document by using the mouse to select objects of interest, then positioning the mouse-driven cursor over one of the menu buttons and clicking a mouse button (an action which we refer to in the text as *buttoning*). Tioga also allows the more common editing operations to be performed on selected objects by using keystrokes from the standard keyboard. The "balloon" surrounding the "B" in "Button" represents a voice annotation.

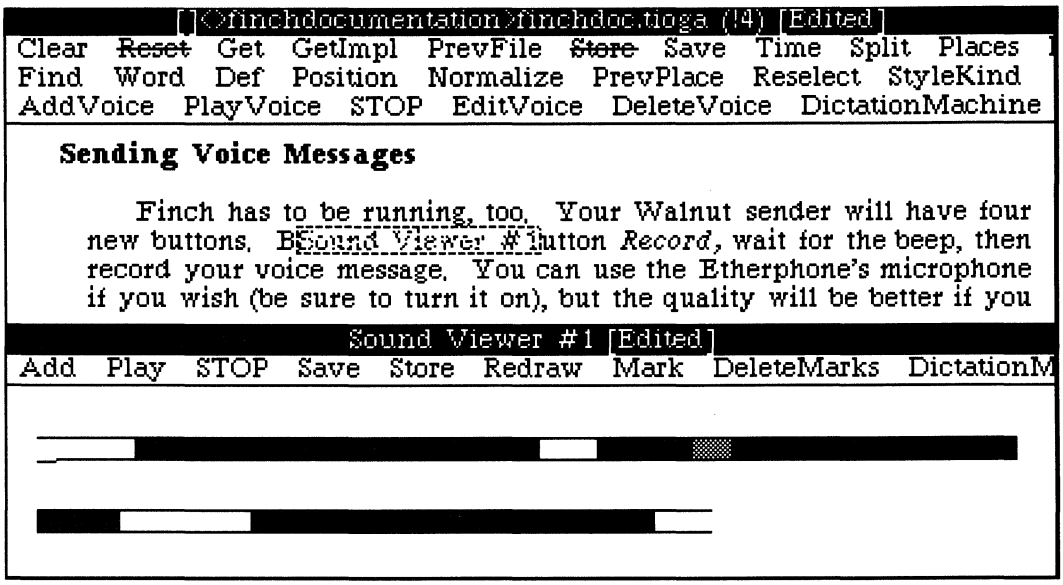


viewer and buttons AddVoice in that viewer's menu. Recording begins immediately, using either a hands-free microphone or the telephone handset, and continues until the user buttons STOP. As recording begins, a distinctive iconic indication of the presence of a voice annotation is displayed as a sort of decoration of the selected character. Currently, this *voice icon* is an outline in the shape of a comic-strip dialog "balloon" surrounding the entire character. The second line of text in the first paragraph shown in Figure 1 contains a voice icon.

Adding a voice icon does not alter the layout of a document in any way. Thus, voice annotations can be used to comment on the content, format, or appearance of formatted text. Moreover, programs such as compilers can read the text, ignoring voice icons just as they ignore font information. Voice annotations may be used, for example, to explain portions of a program text without affecting the ability to compile it. Like font information, voice icons are copied along with the text they annotate when editing operations move or copy that text to other locations, either within the same document or from one document to another.

A voice annotation becomes a permanent part of a Tioga document. Copies of the document may be stored on shared file servers or sent directly to other users as electronic mail. To listen to voice, a user selects a region containing one or more voice icons and buttons PlayVoice. Since playback takes some time, the user may queue up additional PlayVoice requests during playback. These will then be played in sequence. The STOP button can be used to halt the whole process at any time.

Figure 2. The voice icon of Figure 1 has been "opened," producing the voice viewer just below it. Unlike ordinary voice icons, open icons do disturb the formatting of the text in which they appear, to make room for the voice viewer number. A voice viewer displays "capillary tubes" filled with regions denoting sound (black) and regions denoting silence (white). The length of each region corresponds to the time required to play it. Each row of tubing in the figure represents about twelve seconds of voice. The small gray rectangle is the playback "cue" (see Section 5.4).



5.2 Voice viewers

The procedure outlined above is fine for short annotations, but for more complex annotations the user will need facilities to edit portions of voice. To keep things simple for rapid annotation, all that appeared in the text was an icon representing a complete voice utterance. To perform editing operations, the user selects a region of text and buttons EditVoice. A *voice viewer* opens up for each voice icon within the selection. Each of the selected voice icons at this point changes its appearance to that of an *opened voice icon* – it now displays a number that identifies the corresponding voice viewer. Each voice viewer is labelled with its number, so that the user can easily see the associations between voice viewers and positions in text viewers. Figure 2 shows what the voice icon of Figure 1 looks like when opened, along with the corresponding voice viewer.

We have already expressed our desire to steer the user toward phrase-level editing and away from editing at a finer grain. For this reason, we have avoided a type of display that many other designers have used [2, 8, 13, 18], which is a graphical profile of the sound energy plotted against time. Such a display encourages people to identify visually the individual consonants, phonemes, or words within voice. Our viewers essentially look like "capillary tubes" – long horizontal rectangular tubes filled in to indicate positions of sound and unshaded where there is enough silence to identify a phrase or sentence break. As with other voice editors, the length of a capillary tube corresponds linearly to the duration of sound that it represents. The appearance of the voice viewers is very similar to the graphical depiction

used in the Sydis VoiceStation [9].

Although this format heightens the tendency to think in terms of phrase-level editing, it is sufficiently uninformative that users will not remember which capillary segment represents which portion of speech. When editing text, one tends to focus in turn on several portions of text, then to locate them again to make a specific edit. We must be able to do the same with voice. This seems to be what has led other designers to the energy-graph approach, although it is only moderately successful as a means of providing context. We have chosen instead to supply a number of marking techniques that we think can provide this context more successfully, without the unwanted side-effect of encouraging the user to focus on an undesirable level of detail when editing.

5.3 Voice editing

How are editing operations accomplished? In Tioga, the user copies text from one position to another by making two selections with the mouse, then pressing a particular key on the keyboard. The uniform editing philosophy says that exactly the same actions should be used to copy voice from one position in a voice viewer to another. In fact, all of the standard Tioga text-editing operations are used also for editing voice – operations such as deleting, moving, copying, or transposing portions of voice within or among voice viewers.

Some operations have analogous rather than identical meanings within voice and text viewers. "Selection granularities" are one example. We have referred several times to "making a selection." Tioga functions are mainly achieved by selecting some region of text, using the mouse, and then indicating a function to be applied to it. For convenient and fast editing, Tioga offers easy selection of different units or granularities of text – single characters, complete words, entire paragraphs, or entire sections. A selection, once made, can be extended to include contiguous units at the same granularity. Now in voice viewers, our smallest units are not characters or words, but phrases or sentences. Therefore, we have set voice selection granularities to phrase-sized units or larger, so that the user is steered toward making edits at an appropriate level. A finer-level selection is available, providing resolution to within about a quarter of a second, but its use is discouraged for ordinary editing. (We remarked above that edits at the phoneme level, if attempted, invariably sound bad.)

Inserting new voice into a voice viewer is analogous to generating new text at a keyboard for conventional documents. The user selects a position within the voice representation, buttons AddVoice, then speaks into the microphone. While the new information is being recorded, a series of arrowhead symbols is inserted into the capillary tube beginning at the insertion point. New arrowhead symbols are added at the proper rate to indicate in terms of the time scale of the capillary display how long the speaker has been talking. Once STOP is invoked, the arrowheads are replaced by the normal capillary display (possibly enhanced visually using the techniques described in Section 5.4, below).

Finally, the user can save a set of voice edits, buttoning Save to preserve them as the new "contents" of the corresponding voice icon (if there was one), or Store to add the voice as an annotation at any selected position in a text viewer. The use of these buttons is analogous to their

interpretation in text viewers to store edited text into permanent files. When the Store button is used, an open voice icon is created at the new position to represent the changed information. In this case, the voice annotation at the original text position (if any) retains its original value; its open icon reverts to the standard "balloon" form.

5.4 Marking voice

How does the user know which portions of the voice to edit, with only black regions in the voice viewer representing "noise" and white ones "silence"? We have developed a number of methods to help the user establish the needed context.

5.4.1 The playback cue

The simplest way to determine what sound is represented by a capillary tube is to listen to it. In a text viewer, the user can select an area of text and play all the voice contained within its voice icons. Similarly, making a selection in a voice viewer and buttoning Play causes all the voice within the selection to be played. A position indicator or *cue* moves along the capillary display, indicating exactly the position of the voice being heard (see Figure 2). A variant of the Play command plays all of the voice in a single voice viewer; we will later describe some additional playback operations that are useful for dictation. Requests can be queued up in any order from voice icons and from voice viewers, but the visual cue appears only when the voice being played was requested from a voice viewer.

Editing is permitted while playback is in progress, so that the user can readily select voice regions for deletion, replacement, or relocation as the cue moves over them and their contents are revealed audibly. Alternatively, the user may mark points of interest for future attention as the cue passes them, using any of the methods described below. "Editing behind the cue" has turned out to be a fast and nearly effortless method for performing simple voice edits.

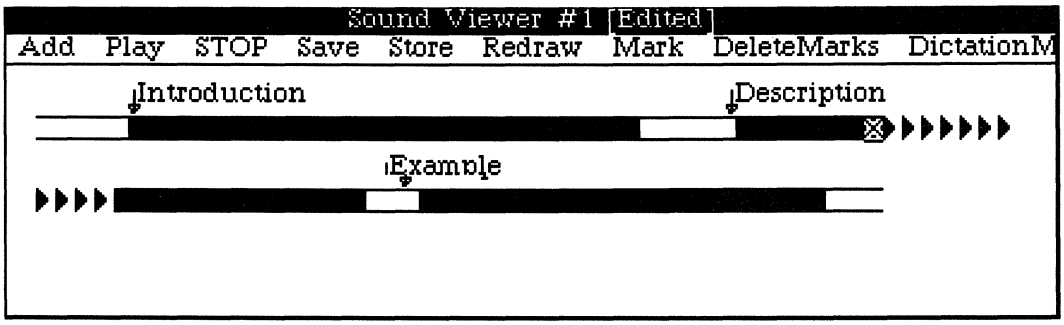
Once a selection of voice has been queued for playback, this playback will proceed, regardless of whether a portion of the queued selection is meanwhile deleted from the voice viewer. This was the most coherent and instinctive behavior we could find for this situation. The cue indicator will disappear during playback of voice segments that are no longer part of the viewer's contents, reappearing when visible voice is reached again.

5.4.2 User-supplied marking

In Tioga, operations such as copying require the user to specify two selections. Making the selections is simple for visual media, but when these operations are applied to voice viewers, it is harder for the user to identify the portions to be selected. If identification involves listening to the voice, then this in turn will involve making selections. There needs to be a way of marking each selection once it has been located.

The user can select any point within a voice viewer and button Mark to place a distinctive diagonal cross within the capillary at that point (and DeleteMarks to remove one). Selecting a range of voice

Figure 3. The voice viewer of Figure 2 has been organized by the addition of permanent annotations, then reordered. The user placed a temporary marker (diagonal cross within the capillary tube) while listening to the previous version, locating the point where new commentary was to be dictated. The arrowheads indicate that recording is still in progress.



rather than a point causes Mark to place a cross at each end of the selection. A mark placed in the middle of a phrase will behave as a new phrase boundary when further selections are made. In this way, user-defined points of interest can be selected as boundaries, in addition to the silence/sound transitions. These marks may be used for any purpose, but a particular intent is to use them as an aid for making multiple selections. These temporary marks are not retained as part of the voice annotation, but are destroyed when the voice viewer containing them is destroyed.

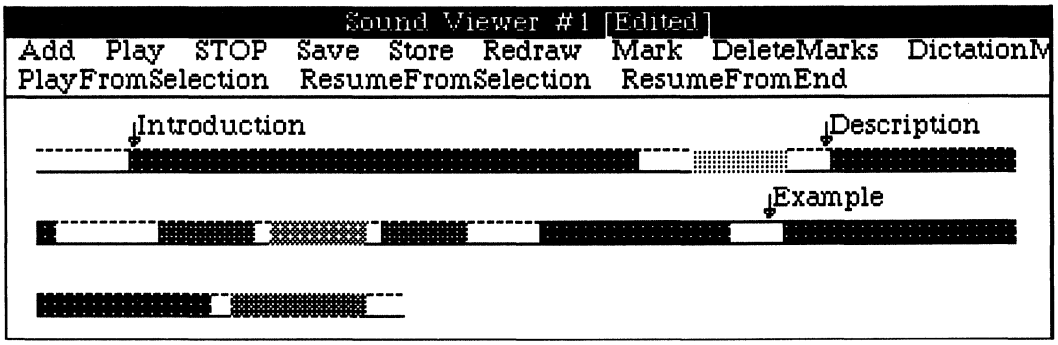
More permanent marks are also available in the form of textual annotations of the recorded voice. Selecting a position in a voice viewer and then typing text causes that text to become associated with the selected voice. The text appears above the selected position, with an arrow pointing at the selected place (see Figure 3). These annotations are retained along with the voice when the viewer contents are saved; they will appear again the next time that voice icon is opened. Very simple editing of these annotations is supported, limited to deletions of single-characters or entire text annotations. Through textual annotation of voice, the user can index any phrase with, for example, some key word or words that occur within the phrase.[4] We believe this kind of textual annotation provides more useful cues for searching within a voice viewer than would an energy profile or other pictorial representation of the voice signal.

5.4.3 "Fresh" versus "stale" voice

Section 3.3 introduced the notion that regions of a voice file that have been inserted or modified recently—"fresh" regions—are different from the remaining, "stale" ones. This difference lies in the ease with which the user can reenter fresh regions. In the Cedar voice editor, we have made novel use of color to test this idea.[5] When a voice viewer is opened, the capillaries representing the voice appear

[4] Automatic speech recognition, however imperfect, might provide such keyword annotations automatically. Occasional mistranslations would be harmless in this context. This concept has been tested experimentally by the MIT Architecture Machine Group [13].

Figure 4. Several edits have been made to the contents of the voice viewer, resulting in this view. Shadings are used to simulate the colors that appear when voice is displayed on a color CRT. The most recently added voice is shown in the lightest shading, representing a bright yellow. Voice unchanged from Figure 3 is shown in the darkest shading, representing a dark red.



in a dark red color. Now suppose that a copying operation is performed. The copied voice appears in its new position in a bright yellow. If another copy is then performed, the most recently copied portion appears in this bright yellow, while the previous copy changes to a less vivid color.

A number of these colors were selected and ordered to suggest an aging process. The most recently added text appears in bright yellow, the next in a deeper orange shade, and so on until all voice of a more than a certain "age" will appear in the dark red. Like the temporary cross-shaped markers, these age indications are a volatile property and disappear from a voice viewer when it is destroyed. In early use of the system, these colorful indicators of recent activity have reduced the need for user-supplied markers. In Figure 4 we have attempted through various shadings to show in black and white the effect of these automatic freshness indicators during editing.

5.5 Dictation functions

5.5.1 Specialized playback and recording functions

In a voice editor, "what you see" is very definitely not "what you get". It is what you *hear* that counts; the display only *represents* what is there. Playing back voice segments is therefore a common operation during voice editing. Dictation of new, "fresh" material has some particular requirements. While dictating, the user may make a mistake or be uncertain whether a word sounded clear. It would be convenient to be able to replay what was just spoken from some point, then to be able to resume recording from either that point ("recording over" what had been there previously), or from the end, if the existing version were satisfactory.

All of this can be done using the Tioga-style voice editing operations that we have described, but

[5] Many Cedar workstations are equipped with a color as well as a monochrome bit-mapped display. On systems with a color display all voice viewers appear by default in color. We will explain below how the voice editing interface differs for users with only the monochrome display.

not always conveniently. We have provided three operations to make it easy to use the editor like a conventional dictation machine. These are accelerators that combine simpler editing operations in specialized ways.[6]

These functions are presented as an additional, optional set of menu buttons (shown as a second row of buttons in Figure 4). Buttoning PlayFromSelection cancels any recording or playback already in progress, then plays all the voice from the current selection to the end of the "freshest" section of voice in the viewer.[7] ResumeFromSelection cancels any activity in progress, deletes the voice from the selection to the end of the freshest section, and then begins recording from the position of the selection. ResumeFromEnd does the same thing, except that it does not delete anything and begins recording from the end of the freshest section.

These three functions are all that are needed to provide a high speed dictation facility. Typically, the user records until he or she makes a mistake or wishes to listen to what has just been dictated. Repeatedly selecting the approximate point with the mouse and then buttoning PlayFromSelection allows the user quickly to find the end of the last phrase to be retained. Then ResumeFromSelection is used to replace the remainder of the previously-recorded segment with new dictation. ResumeFromEnd would be chosen, instead, if the replayed passage turned out to be acceptable.

5.5.2 The DictationMachine Button

The DictationMachine menu button, available in both text or voice viewers, is an accelerator that makes it easier to begin a dictation (for example, to dictate a letter that a secretary will later type). Buttoning DictationMachine produces an empty voice viewer and immediately begins voice recording. The characteristic recording arrowheads appear to indicate progress. Once created, the viewer behaves just like any other voice viewer, except that it is not associated with an open voice icon until the user specifies a location by buttoning Store.

Normally, simple annotations can be entered as described in Section 6.1 without the need for voice viewers or editing operations. However, if during input a mistake is made, the user will find the dictation functions useful. Instead of stopping the recording and then opening a voice window, the user can button DictationMachine. This produces a new voice viewer, as above, whose only contents are the arrows representing the voice already recorded. Recording continues.

We have already described how dictation into an existing voice viewer is performed: the "fresh" voice is distinguished from older segments by color so that selection for editing is easy. Furthermore, the dictation functions automatically select the end of the fresh segment.

---

[6] Tioga includes many other such accelerators and also provides ways for users to define their own.

[7] More precisely, the region chosen by this and the other two specialized dictation operations is from the current selection to the beginning of the next segment less "fresh" than the one containing the selection, or to the end if there is no such segment. It is intended that the end of the freshest segment be chosen, notably the one just dictated.

5.5.3 Dictation on a monochrome display

In order to use the "freshness" ideas with a monochrome display, we could substitute different shadings and stipple-patterns for the colors on the color display. In fact, the Cedar device-independent graphics package will do this automatically if the information to be displayed includes color specifications. However, these patterns may be difficult to distinguish from each other or to interpret as indications of the "degree of freshness." In addition, these shadings will confuse the viewer if combined with the playback cue, cross markers, and the indication of the current selection. Therefore, we have eliminated the freshness markings when using the monochrome display. Newly-added voice fades immediately into its surroundings. The three specialized dictation functions always treat the voice viewer as a single segment of constant age, operating between the selection and the end of the entire viewer.

The DictationMachine button can be used to prevent new material from merging indistinguishably with the old when the user of a monochrome display stops dictation to correct a mistake. Buttoning DictationMachine while recording into a voice viewer transfers the expanding set of arrowheads representing the new material into a newly-created voice viewer. At the insertion point in the original voice viewer a simple (i.e., diagonal cross) marker is left, so that the new voice can be returned (by hand) to that point once the dictation session is complete. This dictation session may include arbitrary edits and additional dictation since the new viewer is in all ways a standard voice viewer. We considered providing an automated command for merging the results back into the original viewer, but we could not find semantics for such a command that had an unambiguous and readily-understood user model. (For example, by repeated use of the DictationMachine button, the user could generate a complex structure of voice viewers to be merged.) In line with the design principles of Section 3, we chose the simpler, manual approach.

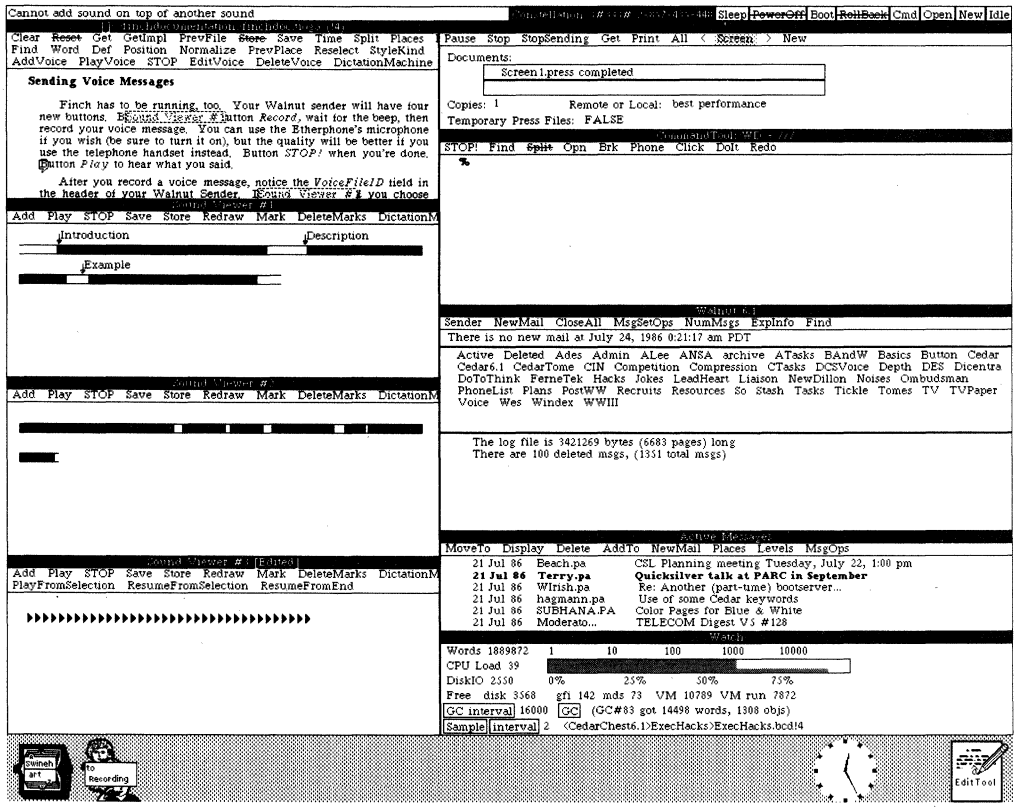
6. Systems Implications

We now discuss some of the implications of the above interface on the system that has to support it. These are mentioned only in outline.

Cedar is an experimental programming environment designed to support rapid development of prototype applications. All of Cedar, and the Tioga editor in particular, was built with extensibility in mind. Tioga provides the mechanism to decorate a text character with arbitrary "artwork," and with additional, invisible, named properties. This made implementation of voice icons straightforward. Cedar includes a number of user interface, screen management, and graphical packages that make the implementation of voice viewers and their editing capabilities a straightforward task.

Recording, playback, editing, and management of recorded voice are enabled by the facilities of the Etherphone system. Voice storage and telephone control services are provided by a Cedar machine running as a central dedicated "server." Users' personal Cedar workstations run a variety of voice-related applications: convenient methods for placing, logging, and managing telephone calls, voice

Figure 5. A voice editing session. A formatted text document is open, along with voice viewers representing two of its annotations. A third voice viewer has been created, using the DictationMachine button, to remove newly-dictated voice from its surroundings for more convenient modification. Viewers for other Cedar applications are also visible on the 13" wide by 10" high screen.



synthesis applications, and various applications for recorded voice, including the document annotation described here. Voice input/output itself is accomplished using a separate, microprocessor-controlled unit called an Etherphone, designed specifically for the voice project. Together, the Etherphones and the telephone control server make a digital, packet-switched, office telephone system, using an Ethernet [21] local area network for both voice transmission and control (an Etherphone has its own Ethernet connection). This system is described in greater detail in [16].

One aspect of the Etherphone system's architecture is particularly relevant to voice editing systems. Digitized voice is stored not by individual workstations, but by a voice file server on the Ethernet, designed specifically for recording and playing voice. The only time the data is directly accessed is to play it back, by sending voice packets from the server back to an Etherphone.

The voice file server records *voice files* directly from Etherphones under the control of workstation applications, creating a unique *identifier* for each new voice file. Applications use these identifiers, rather than handling the actual voice samples, to manipulate voice files and to refer to them in documents. There are a number of reasons for this organization. The primary reason is that voice files

are large – 8,000 bytes for every second of stored voice. We wish to avoid the time required to copy all of the voice included in a document onto the workstation of every user who accesses it, particularly because voice samples must in any case be transmitted to the physically separate Etherphone for playback. Similarly, we avoid the necessity of copying the contents of voice annotations when the documents that they annotate are copied. With improved compression of voice and reduced data storage costs, the argument becomes less strong: the admittedly simpler method of including the voice samples directly in documents might be preferable. However, we look upon such improvements in technology as enabling the inclusion of higher quality audio or even real-time video within documents. The need for a reference-based scheme then remains.

7. Future Trends and Further Work

7.1 Incorporation of speech recognition

Useful lessons can be learned by examining how conventional dictation machines are used. Executives use the dictation machine as an input device because they can speak faster than they can write or type. Dictation is given to a secretary to type up and no attempt is made to play the whole thing back before then, simply because one can read faster than one can listen to voice. Dictations on conventional machines often include editing directives such as "in the first paragraph about artichokes, please change "would like you to" to "urgently require you to," and so forth. The rather more manipulable approach to dictation set out in this paper should encourage more direct editing of the voice instead of such postfixed amendments.

For dictation applications, voice editing tools as described in this paper may eventually become obsolete. As voice recognition systems become more sophisticated and, in particular, when they become generally applicable and affordable enough for inclusion in the personal workstation environment, we expect that the entry of voice, translated instantaneously into text for fast reading, will become a normal mode of use. Multimedia editors may well then become integrated to the extent that alterations to textual portions of a document may be done by vocal input.

Accurate translation of continuous speech into text is unlikely to be possible for some time. However, systems that instantly translate dictation into reasonably-accurate text that the user can then correct should be available much sooner. A secretary using such a system to transcribe other people's dictations would benefit from the ability to listen to the original voice while correcting the automatic translation. However, there are some uses of voice annotation that would not benefit from a translation to text. Inflection, emphasis, and the personal touch of a familiar voice are all part of the value of a spoken annotation.

7.2 Extensions of the annotation concept

While the incorporation of accurate speech recognition into a multi-media document system is still far from being a reality, the concept helps highlight the need for a more generalized document

structure and editing interface than we have set out here. There are several reasons for wanting to extend document architectures beyond what has yet been accomplished.

For example, we have limited ourselves to vocal annotation of documents, with simple textual annotation of the voice but no further nesting of structure. There are three obvious directions in which we might extend these ideas. First, there is no reason to restrict annotation to vocal annotation. Just as a voice viewer can be "popped up" by selecting and manipulating a voice icon, a visual entity could be "popped up" and manipulated in an analogous way. This might be a textual annotation of a textual document, analogous to "scribbling in the margin." Annotations using computer animation and video are also possibilities. Secondly, there is no need to restrict the kinds of information that can be annotated. For example, voice annotation of illustrations and scanned objects could be as useful as annotated text. Extending in a different direction, it might be useful to apply annotations to a range of text or other objects, if a clear way could be found to indicate the presence of overlapping annotations to the user. Finally, there is no need to restrict annotation to a fixed number of levels. For example, an annotation hierarchy, with voice annotating the text that annotates a video commentary of a multi-media document, makes quite good sense. An on-line documentation system for computer systems is an application where a complex system of documents annotating one another could be used to help guide the user quickly to the desired information. The general model is that documents can be nested within other documents, either by direct inclusion or by annotation.

We have also considered higher-level applications that would use annotation as a component. We suggested in the introduction an advanced use of voice (and potentially video) annotation, in which a document would become an automatically-narrated audio-visual presentation. The document would contain timing and sequencing information to scroll specified areas into view on cue as the recorded narrative paced the presentation.

Most of the extensions suggested here could be accommodated within the framework of existing uniform user interfaces – the user would not have to learn large numbers of new concepts. We do expect, however, that each new medium will introduce the need for a few specialized concepts and operations, as we discovered in the present work. For instance, the behavior of a voice annotation still needs to be slightly different from that of other "pop-up" annotations, because there is a need for both a simple playback capability and the more heavyweight opening of a voice viewer to edit the voice. There is no direct parallel to these two separate requirements in the case of visual media.

7.3 System management of other media

As we indicated in Section 6, we consider it impractical to include digitized voice directly in annotated documents because of its size, choosing instead to store references to the voice. The inclusion of scanned images into documents introduces another medium that, like voice, involves a substantial amount of data that will be manipulated in large units. This suggests that such images might also be represented in documents by embedded references, to reduce the time and storage

required to copy the documents. Real-time video annotations and cross-references among documents could also be implemented using this approach.

8. Conclusion

In this paper, we have described the user interface for a voice annotation and editing system. The key points of our design are:

- Voice is treated as an additional medium to be incorporated into a multi-media document management system. The voice facilities have been added by extending the semantics of an existing user interface to encompass voice where appropriate, then by adding new techniques to deal with the idiosyncrasies of the audio medium.
- There are some cases where simply converting the semantics of a text editing interface to voice would yield poor results. In such cases, we have produced a deliberately different interface. For example, we restrict voice editing to the manipulation of quantities no smaller than a spoken phrase, using a very simple capillary representation of the phrase structure. We have concluded that more elaborate energy profile representations stress too fine a level of detail, and may provide more distraction than contextual information.
- This prototype voice editor only required two months to implement. This was possible because the components of the Cedar programming environment were designed to be extensible. The editor was able to directly use a number of user interface facilities already available in the environment. The Etherphone system supplied the underlying capabilities for telephone control as well as for recording, playback, and low-level voice-editing operations. Extensions were linked into Tioga to add voice icons and the specialized voice recording, playback, and dictation commands.

We have just begun to test this voice editor within the Cedar community. We will discover which aspects of our design find favor with users and which need improvement. There are many ways in which this work could be extended, some of which have been outlined above. We believe that future work should continue our efforts to balance the need for a user interface that is easy to understand and easy to use against the desire for an extensible and general structure that enables fluent and efficient manipulation of a variety of media.

Acknowledgments

The authors would like to thank Polle Zellweger and Doug Terry, both members of the Etherphone project, for the vital roles they played in the design and implementation of the voice editor, and for their valuable comments and encouragement during the preparation of this paper. Michael Plass deserves special mention as a consultant and an "in-the-nick-of-time" implementor of the Tioga features that made this work possible.

References

[1] R. Allen. "Composition and Editing of Spoken Letters," *International Journal of Man-Machine Studies* 19 (2), August 1983, 181-193.

[2] The Centerpoint System, produced by Santa Barbara Laboratories, Inc., of California. Descriptions are based on observations by the authors of the system in use.

[3] H. Forsdick, R. Thomas, G. Robertson, and V. Travers. "Initial Experience with Multimedia Documents in Diamond," *Proc. IFIP 6.5 Working Conference*, May 1984, 97-112.

[4] W. Horak, "Office Document Architecture and Office Document Interchange Formats: Current Status of International Standardization," *IEEE Computer*, October 1985, 50-59.

[5] J. Garcia-Luna, A. Poggio, D. Elliot. "Research into Multimedia Message System Architecture," *Report on SRI Project 5363*, February 1984.

[6] N. Maxemchuk. "An Experimental Speech Storage and Editing Facility," *Bell System Technical Journal* 59(8), October 1980, 1383-1395.

[7] D. McCracken and R. Akscyn. "Experience with the ZOG human-computer interface system," *International Journal of Man-Machine Studies* 21(4), October 1984, 293-310.

[8] B. Mirrer. "An Interactive, Graphical, Touch Orientated Speech Editor," Masters Dissertation, Laboratory for Computer Science, MIT, 1982.

[9] R. Nicholson. "Integrating Voice in the Office World," *BYTE*, December 1983, 177-184.

[10] R. Nicholson. "Usage Patterns in an Integrated Voice and Data Communications System," *Transactions on Office Information Systems* 3(3), July 1985, 307-314.

[11] J. Reynolds, J. Postel, A. Katz, G. Finn, A. DeSchon. "The DARPA Experimental Multimedia Mail System," *IEEE Computer*, October 1985, 82-89.

[12] A. Ruiz. "Voice and Telephony Applications for the Office Workstation," *Proc. 1st International Conference on Computer Workstations*, San Jose, CA, November 1985, 158-163.

[13] C. Schmandt. "The Intelligent Ear: A Graphical Interface to Digital Audio," *Proc. IEEE Conference on Cybernetics and Society*, October 1981, 393-397.

[14] C. Schmandt and B. Arons. "Phone Slave: A Graphical Telecommunications Interface," *Proc. Society for Information Display 1984 International Symposium*, June 1984.

[15] D. Smith, R. Kimball, B. Verplank, E. Harslem. "Designing the Star User Interface," *Byte* 7(4), April 1982, 242-282.

[16] D. Swinehart, L. Stewart, and S. Ornstein. "Adding Voice to an Office Computer Network," *Proc. of GlobeCom 83, IEEE Communications Society Conference*, Nov. 1983.

[17] D. Swinehart, P. Zellweger, and R. Hagmann. "The Structure of Cedar," *Proc. of ACM SIGPLAN 85 Symposium on Programming Languages and Programming Environments,"* June 1985.

[18] R. Thomas, H. Forsdick, T. Crowley, R. Schaaf, R. Tomlinson, V. Travers, G. Robertson. "Diamond: A Multimedia Message System Built Upon a Distributed Architecture," *IEEE Computer*, December 1985, 65-77.

[19] Wang Laboratories, Inc. "Audio Workstation Author Guide," 1st Edition, 700-6989, 1982.

[20] H. Wilder and N. Maxemchuk. "Virtual Editing II: the User Interface," *Proc. of the SIGOA Conference on Office Automation Systems*, Philadelphia, Penn. 1982.

[21] Xerox Corporation, Intel Corporation, Digital Equipment Corporation. "The Ethernet: A Local Area Network; Data Link Layer and Physical Layer Specifications," Version 2.0, November 1982.

XEROX

Voice Annotation and Editing in a Workstation Environment Stephen Ades and Daniel C. Swinehart

CSL-86-3