[Fox91] Jackie Fox. Unlocking the door: Pcs and people with disabilities. *PCToday*, pages 43-51, March 1991.

[Gav89] William W. Gaver. The SonicFinder: An interface that uses auditory icons. *Human Computer Interaction*, 4:67-94, 1989.

[KY87] Richard M. Kane and Matthew Yuschik. A case example of human factors in product definition: Needs finding for a voice output workstation for the blind. In *Proceedings of ACM CHI+GI'87 Conference on Human Factors in Computing Systems and Graphics Interface*, eds. John M. Carroll and Peter P. Tanner (Toronto, Canada: ACM, 1987) pages 69-73.

[Opp83] Alan Oppenhiem. *Signals and Systems*. Prentice-Hall, 1983.

[OS75] Alan Oppenhiem and Schafer. *Digital Signal Processing*. Prentice-Hall, 1975.

[Ros90] Thomas D. Rossing. *The Science of Sound*. Addison-Wesley, 1990.

[SBJG86] Denise A. Sumikawa, Meera M. Blattner, K. Joy, and R. Greenberg. Guidelines for the Syntactic Design of Audio Cues in Computer Interfaces. Technical Report No. UCRL 92925. Lawrence Livermore National Laboratory, 1986.

[Van89] G.C. Vanderheiden. Nonvisual alternative display techniques for output from graphics-based computers. *Journal of Visual Impairment and Blindness*, pages 383-390, vol. 83, October, 1989.

[Wol88] Stephen Wolfram. *Mathematica*. Addison-Wesley, 1988.

[WWF88] E.M. Wenzel, F.L. Wightman, and S.H. Foster. A virtual display system for conveying three-dimensional acoustic information. *Proceedings of the Human Factors Society, 32nd Annual Meeting*, (Anaheim, California: HFS, 1988), pages 86-90.

REFERENCES

[Ant79] Andreas Antonious. *Digital* Filters*: Analysis and Design*. McGraw-Hill, 1979.

[Bla73] Jens Blauert. *Spatial Hearing*. MIT Press, 1973.

[Bly82] Sara Bly. Presenting information in sound. In *Proceedings of Human Factors in Computer Systems*, ed. Michael Schneider (Gaithersburg, Maryland: ACM, 1982) pages 371--375.

[BGB91] Bill Buxton, Bill Gaver, and Sara Bly. Using Nonspeech Audio at the Interface/ Tutorial presented at CHI '91. April 1991.

[BBV90] L.H. Boyd, W.L. Boyd, and G.C. Vanderheiden. The graphical user interface: Crisis, danger and opportunity. *Journal of Visual Impairment and Blindness*, pages 496-502, December 1990.

[BBV91] L.H. Boyd, W.L. Boyd, and G.C. Vanderheiden. Graphics-based computers and the blind: Riding the tides of change. In *CSUN Technology and Persons with Disabilitie*s *Conference Proceedings*, 1991.

[Bur92] David Burgess. Low cost sound spatialization. In *User Interface Software and Technology Conference Proceedings*. November 1992.

[Bux86] William Buxton. Human interface design and the handicapped user. In *CHI '86 Conference Proceedings*, pages 291-297, 1986.

[CH87] Stuart K. Card and D. Austin Henderson, Jr. A multiple, virtual-workspace interface to support user task switching. In *Proceedings of ACM CHI+GI'87 Conference on Human Factors in Computing Systems and Graphics Interface*, eds. John M. Carroll and Peter P. Tanner (Toronto, Canada: ACM, 1987) pages 53-59.

[CMK88] John M. Carroll, Robert L. Mack, and Wendy A. Kellogg. Interface Metaphors and User Interface Design. *Handbook of Human-Computer Interaction*, M. Helaander (ed), Ebevier Science Publishers B.V. (North-Holland), pages 67-85, 1988.

[Che53] Some experiments on the recognition of speech with one and two ears. *Journal of the Acoustical Society of America***,** 22, 61-62.

[HC86] D. Austin Henderson Jr. and Stuart K. Card. Rooms: The use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Transactions on Graphics*, 5, No. 3 (July 1986), pages 211-243.

[FVFH90] James Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics, Principles and Practices*, Addison-Wesley Publishing Co., 1990.

Our research up to this point has resulted in the Audio Rooms metaphor presented in this chapter. Of course, much user testing and evaluation will be necessary to validate or reject the metaphor and variations on its implementation. We plan to conduct extensive user testing.

During the course of the project we are conducting research into the generation and use of spatialized audio. We are now able to generate high-quality spatialized audio in real time using current-generation workstations. Further research concerns the use of everyday sounds in purely auditory interfaces.

The Mercator project is being conducted at the Georgia Institute of Technology and is funded by Sun Microsystems and the NASA Marshall Space Flight Center (Grant # NAG8-194). In addition to the two authors, other people working on the Mercator project include Ian Smith, Dave Burgess and Tom Rodriguez from the College of Computing and John Goldthwaite from the Center for Rehabilitation Technology.

- Head shadowing effects--some sound components are dampened as a lateral sound passes through the head and upper torso.
- Pinnae effects--the human pinnae (outer ear structures) impose a unique signal “fingerprint” on a sound source at a given azimuth and elevation.

These effects are fairly well understood and may be synthesized by a computer [Bla73][Ros90][Ant79][Opp83][OS75][Wol88]. The chief problem currently is the computational time required to perform the spatialization. Current hardware solutions (such as the Crystal River Convolvatron) are quite expensive [WWF88]. In the near future, however, software solutions will be possible on general purpose computers howeve [Bur92]r.

How will this synthetic spatialization be used in the interface? We feel that the use of spatialized audio would be most beneficial in aiding users to determine the layout of objects in rooms. Just as in a real room, each object in the room has a particular location, in Audio Rooms each application or file has a particular location. Spatialized audio can be employed to inform the users of the location of objects relative to his or her position.

Note that this technique is quite similar to virtual reality systems in which users are, in effect, placed “inside” a computer-generated environment [FVFH90]. In Audio Rooms, users are free to wander through their rooms layout and manipulate objects at will. The environment should sound similar to an actual rooms environment.

7.0 The Mercator Project

Many of the ideas described above are currently being implemented in a system called Mercator by the Georgia Tech Multimedia Computing Group. The goals of the system are two-fold:

- Provide an elegant, powerful environment for navigating through the computer system and its applications and files.
- Provide access to existing graphical applications.

The first goal is being addressed using the Audio Rooms approach described in this chapter. The second goal, which has not been addressed here, will result in the construction of a system for retrieving and presenting relatively high-level semantic structures from applications.

Our platform for the project is Unix workstations running the X Window System. We believe that the system, when finished, will allow users with visual disabilities to work using an environment that is every bit as natural and productive as the graphical systems used by their sighted coworkers. Furthermore, we hope to provide access to existing X applications so that all members of a workgroup, both sighted and non-sighted, can use the same tools to accomplish their work.

6.2.2 Everyday Sounds

Computers and other man-made devices make a variety of beeps, buzzes and other artificial noises that otherwise we would never hear. Conversely, we typically hear things crumbling, things sloshing, things colliding - we hear the results of objects interacting together. When asked to describe a sound, we tend to describe it in terms of the objects that generated the sound such as a door slamming, stairs creaking or glass breaking. In other words, we describe sounds in terms of their source not in classical terms such as pitch and duration.

What this realization means to an interface designer is that we can use sounds to make users think of familiar objects in the same way that icons in GUIs characterize commonplace objects. Gaver first introduced the term auditory icon with his interface called SonicFinder. [Gav89] SonicFinder added auditory cues to the graphical Finder environment on the Macintosh. Dragging a file icon with the mouse sounded like dragging something along the floor. Dropping an icon into the trashcan sounded like dropping a large object into a metal trashcan.

These concepts can be exploited to their fullest in the Audio Rooms environment. Going through a door can sound like opening a slightly creaky door. A copying operation can sound like a copying machine. Likewise, when the user sends a file to the printer, the sound of a laser printer will be heard. And when the printing is completed, the sound will stop, creating an unobtrusive notification that the print job is complete.

Some sounds will be artificial. A computer application does not have an associated everyday sound, but it can have a standard base sound such as a metallic object. Each room can also have a distinctive sound that is partially determined by the size of the room.

The point is to create a recognizable environment with the use of common, symbolic sounds. One problem remains, how to sort through these sounds so that the interface is more than a cacophony. The next section discusses that solution.

6.2.3 Spatialized Audio

In everyday life, humans perceive sounds in their environment as emanating from some source in space. The ability to determine the position of the sound source in relation to the listener is referred to as localization. It is in large part the ability to localize sounds which enables us to parse a volume of different sound sources and attend to individual sources as desired.

It is possible to exploit localization abilities in the interface. The process of filtering an audio signal to introduce synthetic localization cues is termed spatialization. These localization cues are largely the result of three effects:

- Interaural time delays--the difference in time between when an audio signal strikes one ear and it strikes the other.

- The vocabulary should be context sensitive. For example phone number should be read as “eight nine four dash three six five eight,” not as eight hundred ninety four hyphen three thousand six hundred fifty eight.” A Internet mailing address should be read as beth at c c dot g a t e c h dot e d u (beth@cc.gatech.edu).

Many of these features are now appearing in software-based speech synthesis systems. Refer to Paul Blenkhorn’s chapter in this book for further information on design criteria for speech synthesis systems.

Synthesized speech is a requirement for any accessible system since computers still primarily convey textual information even with GUIs. But speech as a form of output has its limitations. Although we can monitor multiple conversations at a time, it is doubtful that we would want multiple applications talking to us at once for the same reason that we do not want multiple people talking to us at once. Speech requires directed attention. Also speech, like text, is bulky. Text and speech are both serial in nature, that is, they must both be processed from beginning to end whereas graphical images are processed as a whole. Furthermore, graphical images are powerful because one picture can be used to replace a lengthy text description. In the same fashion, non-speech audio cues can replace long streams of synthesized speech. In the next section, we will examine creative uses of non-speech audio to convey large amount of information in single sounds and to create a computer environment that embodies the metaphor of Audio Rooms.

6.2 Non-Speech Audio

6.2.1 Multidimensional Sound and Auditory Motifs

The majority of work with non-speech audio has focused on the manipulation of classical dimensions of sound such as pitch and volume and the use of auditory motifs to carry information. Information is mapped onto different properties of sounds. Mapping information onto the different dimensions of sounds has been used with some success for multivariate data analysis. Sounds can carry information even when a graphical representation of the same data would be impossible to comprehend or even produce. Most studies have found changes in pitch to be the easiest to perceive. [BGB91][Bly82]

Blattner introduced the term “earcon” for auditory motifs that can be used to carry information. Just as each character in “Peter and the Wolf” had a distinguishing motif, different sources of information in a computer application could have recognizable motifs. Also different types of messages could follow different patterns in pitch and rhythm. [SBJG86]

Although auditory motifs are intriguing, in our implementation of the Audio Rooms metaphor we will concentrate on the manipulation of dimensions of sounds to convey some forms of information. But what sounds will we use? The next section introduces the concept of everyday sounds and their use as auditory icons.

6.0 Sound

The Audio Rooms environment is designed for users whose primary means of gathering information is hearing. Hearing is a powerful perceptual process. We constantly monitor our environment by listening to the sounds around us. For example, right now I can hear the hum of my computer, the dishwasher running in the kitchen, my dog walking down the hallway and my colleague talking on the phone. Often we are not even aware of the constant stream of auditory information that we are processing. Due to our perceptual abilities, hearing is naturally multitasking. For example, we can monitor multiple conversations at the same time. This phenomenon is known as the "cocktail party effect." [Che53] In contrast, vision could be thought of as a focused single-task process that relies on fast context-switching to compile complex information. We gather information about what we are currently looking at, and we can only look at one place at a time. For these reasons, we believe that sound can capture the multitasking properties of computer systems as well or better than graphical output. For example, many graphical interfaces signal the appearance of a dialog box or message with an annoying beep in case the user does not notice its appearance on the screen.

Computer generated sounds can be divided into two classes: synthesized speech and nonspeech audio. Current systems for computer access for blind users are dominated by the use of synthesized speech. We will briefly discuss the features and difficulties of using synthesized speech. We believe that the use of nonspeech audio can provide a much richer and powerful interface in the same way that graphics can convey more information than text.

During this section, we will provide examples of sound currently used in computer applications. We will also discuss how we plan to employ these same strategies to create an intuitive Audio Rooms environment.

6.1 Speech

The invention of computer synthesized speech made it possible for blind computer users to interact with computers in a reasonably fast and efficient manner. We believe designers using synthesized speech should strive to meet the following requirements:

- The speech should be understandable and a large vocabulary should be supported.
- Variable speed is necessary for scanning information at a rate much faster than a natural speaking pace.
- Different voices and inflections should be provided. A variety of voices can differentiate different types of messages in the same way that color and other graphical methods add meaning to textual information. The use of inflection patterns can signal that more textual information is available, or that the computer is waiting for some input, or that the computer is informing the user of a serious malfunction.

Characteristic	**Match**	**Subset Miss**	**Superset Miss**
PHYSICAL			
Shape		X	
Size	X		
(Dynamic size)			X
Closets	X		
Windows		X	
Acoustic ambiance	X		
Floor coverings, color, wallpaper		X	
OBJECTS			
Manipulate	X		
3D layout		X	
Permanence	X		
Same object in multiple rooms			X
User specified placement	X		
NAVIGATION			
Doors	X		
Hallways		X	
Layout	X		
Dynamic Layout			X
Teleports			X
USER			
Located in one and only one room	X		
Can move between rooms	X		
Can carry objects between rooms	X		
Can hear events in other rooms	X		
Rooms can contain more than one person	*[a]		
OTHER			
Names	X		
Functionality	X		
Configuration	X		

TABLE 2. Comparing Real Rooms and Audio Rooms

a. Although not included as part of our initial design, a multi-user version of Audio Rooms would support the notion of multiple people in one room perhaps collaborating or working independently of each other.

attempts to model. Usually these superset misses are added to bring about a more usable and powerful interface.

Audio Rooms is enhanced by the addition of several of these superset traits. Most of these traits are also present in the graphical rooms implementations and research has shown them to be useful. For example, in Audio Rooms it is possible for the same object to be present in more than one room. A useful example of this feature would be a new-mail indicator program which would be present in each room. Thus the user could be in any room in the system and be informed when new mail arrives.

Another example is the ability to directly “teleport” from one room to another. In some circumstances it may be annoying to have to maneuver through a series of rooms to reach a desired destination. While the concept of rooms maintains that users will typically install doors between rooms in order to move between rooms, there is still the need to provide a mechanism for instantaneous transportation to a new location. It should be noted that while most of us do not experience the teleporting phenomenon in our own lives, the concept is still easily understandable to most people. The idea of instantaneous transportation is common in both science fiction movies and video games and its inclusion in the rooms metaphor does not disrupt the model to the point of making it unusable.

Two other traits are present in Audio Rooms which are not found in our daily experience. First, the rooms system is easily extendable by its users. A user may choose to add a new room to support some type of activity, or to hold the work related to a particular project. The user may then establish doorways from this new room to chosen existing rooms. In real life, systems of rooms are not so easily extensible. Also, the Audio Rooms environment provides a more complex system of user customization options than real rooms do (such as changing the acoustic properties of a room, enabling the automatic enunciation of a room’s name upon entering, and so forth).

At this point, we have not identified any direct conflicts between the rooms metaphor and the behavior of our environment. User testing will undoubtedly uncover some conflicts. In general we feel that rooms provides a good interface metaphor which has useful traits for interfaces at large. The following table contains the characteristics of rooms which we identified earlier. Each characteristic is labeled as a match, subset miss, or superset miss.

So far we have said little about how we plan to implement the Audio Rooms metaphor. In the following section, we will discuss the various attributes of sound which will be important in an actual implementation of the metaphor as a computer interface.

(or the same room). There may potentially be many people in a given room. This last trait raises interesting possibilities of collaboration between users in a distributed Rooms space. Both sighted and unsighted users could collaborate through their respective rooms implementations.

Like real rooms, each AudioRoom will have a name associated with it. Users assign names to rooms and may refer to them later via these names. Users are free to use a given room for whatever purpose they deem them most fit. The system itself does not enforce a certain functionality to be associated with a given room however. Typically, as in GUI rooms, users will create a room for a given purpose (such as an editing room, or a mail reading room), and then place objects within that room which support that purpose.

5.2.2 Trait Misses

Trait misses are traits which are either present in real rooms but are not present in Audio Rooms, or traits which are present in Audio Rooms but not real rooms. We classify the two types of trait misses as either subset misses (meaning that the trait is present in the metaphor but not the implementation) or superset misses (meaning that the trait is present in the implementation but not the metaphor).

First we deal with subset misses. There are several characteristics of real rooms which either do not map well into our interface, or which basically serve no purpose. For example, we have eliminated the concept of windows in walls as the ability to view “outside” the rooms environment does not have any meaning. Similarly, we can safely ignore room characteristics such as floor covering and wall paper. In a visual system these traits may serve to give visual cues as to the room’s identity, but in an auditory system they are useless. Auditory cues such as the acoustic ambiance of the room will aid the user in identifying a room.

We also chose to ignore the shape characteristics which real rooms may have. In Audio Rooms all rooms have the same shape, primarily for ease of navigation within a room. Room shape does play a role in the acoustic properties of a room, but for our purposes size will serve as the sole acoustic determiner.

Within a real room, objects may be laid out in three dimensional space (a painting may be hanging on the wall directly over a sofa). In Audio Rooms room contents are restricted to being laid out in a two-dimensional plane. Again, this is primarily for ease of navigation and object selection within a room. There are few good three dimensional input devices and those that do exist are quite expensive.

Finally, Audio Rooms does not provide the concept of hallways. Doors are the sole means of transferring locality from one room to another. In our environment, hallways would be essentially an entity with the characteristics of both rooms and doors. We felt that the interface metaphor could be simplified by omitting hallways.

There are also several superset misses. For various reasons, it is often useful to include traits in an interface which are not present in the real world system which the interface

Physical	Objects	Navigation	User	Other
Shape (quality)	Manipulate (action)	Doors (attribute)	Located in one and only one room (quality)	Names (attribute)
Size (quality)	3D layout (quality)	Hallways (attribute)	Can move between rooms (action)	Functionality (attribute)
Closet (attribute)	Permanence (quality)	Layout (quality)	Can carry objects between rooms (action)	Configuration (action)
Windows (attribute)			Can hear events in other rooms (quality)	
Acoustic Ambiance (quality)			Rooms can contain more than one person (quality)	
Floor covering, color, wallpaper (attribute)				

TABLE 1. Characteristics of Real Rooms

5.2.1 Trait Matches

There are many characteristics of rooms which map well to a computer interface, which leads us to believe that the rooms metaphor is a good one. A metaphor which did not exhibit a significant set of attributes which carried over to the interface would likely be a poor choice.

For example, from the physical category, our audio rooms will have closets which can be used to store objects that are not needed on an everyday basis. We will also keep the size attribute associated with our rooms. Unlike real rooms however, we will modify the size of rooms dynamically to indicate the quantity of objects within the room. While this dynamic sizing is not exhibited in real rooms, we feel that it is a valuable trait which will enhance the usability of the interface. (We shall examine “trait misses” in more detail in a moment.)

Each room may contain objects, and each object in a room has a certain absolute location within the room. The objects in a room may be manipulated by the users of the system and, like real rooms, the objects display general object permanence. The placement of objects within a room is left up to the personal preferences of the user and is not decided by the system. Thus users are free to organize their personal workspaces as they see fit.

Rooms will have doors which will connect the rooms to each other. Users transition between rooms via doors. Like the real world, users can be in only one room at a time, they are free to move objects from room to room, and they can hear activity in other rooms closets. Additionally rooms have certain visual and auditory characteristics. A certain room may have a particular type of floor covering or wall paper, and may present certain acoustic properties.

Rooms usually contain objects which the occupants of the room are free to manipulate (move, activate, and so on). These objects are usually placed in a room in some three dimensional layout according to the preferences of the occupants or owner of the room. Furthermore, we can usually expect “object permanence” within a room. That is, when we leave an object in a room we can expect it to be there and in the same position the next time we return.

How do we navigate among real rooms? All rooms have doors in them which usually lead to other rooms. In addition hallways may serve to connect rooms, but they themselves may contain contents as a “normal” room might. The layout of rooms is static and has a “flat” organization, that is, we do not usually think of rooms containing other rooms.

People themselves can be in only one room at a time, although there can be more than one person in a room at a time. A person may be able to hear events which occur in other rooms nearby (for example, a chandelier crashing to the ground). People are free to move objects from room to room.

Most rooms have a name associated with them and are intended to serve some specific purpose (such as a kitchen is for food preparation). Rooms have a general configuration consisting of the contents of the room and the placement of those contents.

The following table summarizes these characteristics into five categories: “Physical” for physical characteristics, “Objects” for the characteristics of objects within a room, “Navigation” for how rooms are connected, “People” for the characteristics associated with people in rooms, and the obligatory “Other” category for those characteristics that do not fit in the earlier categories yet do not form an identifiable grouping. Each characteristic is labeled as a quality of a room, an attribute associated with a room, or an action possible in a room or group of rooms. Qualities determine the general behavior or rules predicted by the metaphor. Attributes are elements predicted by the metaphor, usually physical objects, and actions are simply actions that the metaphor predicts that a user should be able to do..

By enumerating the well-known characteristics of rooms it is hoped that we can develop a non-graphical implementation of the rooms metaphor which will provide the same power and elegance to non-sighted users that the graphical rooms implementations provide to sighted users. We shall now examine the individual room characteristics to determine which are useful to us. As we shall see, some characteristics will map easily and sensibly to our proposed environment; others will not. Additionally it may be necessary to “jump outside the metaphor” to include some characteristics in the interface which are not a part of real rooms.

5.1 A Good Starting Point

The concept of rooms was first developed by the Xerox Palo Alto Research Center (PARC).[HC86] The notion arose out of evaluations of the work habits of people using GUI-based systems. Observers noticed that people tended to group windows based on functionality. In general, transitions would be made between groups of windows during a session with the computer.

In the PARC system, which is graphical in nature, rooms provide higher-level organizational entities into which application windows are grouped. Each group of windows is said to exist in a virtual room; the contents of the room are the windows in the group associated with the room. There are connections ("doorways") between rooms through which users may transition from one room to another. These connections approximate the transitions which users make during a typical work session. [CH87]

Since the initial Xerox PARC Rooms implementation, multiple implementations on top of several different GUIs have been developed. The metaphor of traveling through a network of rooms into which users have grouped windows has been shown to be a powerful organizational device. Rooms-like systems are just now beginning to become available for commercial systems (Hewlett-Packard's Visual User Environment^{tm}, for example).

In their current graphical implementations, rooms systems provide a higher level of conceptual organization over simple windows. Windows serve to group information related to a single application or task; rooms serve to group applications into related sets. It is our belief, however, that the rooms metaphor has additional benefits which can help create a powerful nonvisual interface for visually disabled users.

A nonvisual implementation of rooms, which we will call Audio Rooms, would, of course, provide the same cognitive organizational benefits Rooms provides to sighted users. In addition there are several characteristics of Audio Rooms which we believe make the system especially attractive to users with visual disabilities. First, the metaphor is well-understood by all users. Additionally, sight-impaired people must, of necessity, have a well-developed sense of spatiality and good spatial memory in order to navigate around their own familiar surroundings, such as their living areas. Second, the rooms metaphor is especially rich. The objects and operations of the metaphor should map particularly well into an audio-based interface, as we shall see in the next section.

5.2 Room Characteristics and Extents of the Metaphor

In this section we shall explore the rooms metaphor in more detail. We shall examine the process of identifying an interface metaphor, determining the components of the metaphor which will be mapped into the interface, and determining the quality of the "match" to our desired system.

To begin our discussion, let us catalog some of the characteristics that come to mind when we think of real physical rooms. First, rooms have certain physical qualities: they have size and shape and occupy some volume of space. Rooms may have windows, doors, and

The flaw in this approach is that it forces a visually-impaired user to try to understand an application interface in terms of its visual presentation. A graphical interface is designed to be processed by the visual system. Simply adding auditory cues to a graphical presentation does not generally create a suitable or equally powerful auditory presentation.

Another approach which has been tried is to simply organize the application windows which are present on the screen into a list or menu. Then users can traverse the list (usually through keyboard interaction) until they select the application they want to work with. Both of these approaches basically do away with all the cognitive benefits which GUIs provide: simultaneously available multiple information sources, direct manipulation of objects, and a rich metaphor to aid in building a mental model of the computing environment. The problem with these approaches is that they attempt to simply retrofit the graphical desktop metaphor with audio information.

We believe that systems designers should take a step back in the interface design process and make use of another metaphor more suited to nonvisual interfaces. We have identified an interface metaphor which we believe is more suitable for a nonvisual implementation. This metaphor, Rooms, has already been implemented in graphical systems [CH87] and it seems to provide substantial benefits. We intend to explore a nonvisual implementation of Rooms, which we call Audio Rooms. We believe that the Rooms metaphor is well suited to the needs of our target community.

In the Rooms metaphor, each room serves as a container for grouping some related applications or data. Users may place objects in these rooms according to their own preferences. In Rooms, each object represents either an executable application, some data or text file, or a doorway which leads to another room.

Users manipulate objects in rooms by first selecting them and then by choosing an operation to perform on the selected object. The operations available for an object depend on the type of the object. For example, doorways can be traversed, while files may be copied, deleted, or renamed. Applications may be started or stopped.

These simple ideas form the core of the Rooms metaphor. Rooms provides mental devices which support cognitive organization of information at a higher level than the desktop metaphor. We shall examine Rooms and Audio Rooms in greater detail in the following section.

5.0 Audio Rooms

We will begin our discussion by describing why we think Audio Rooms could be a good metaphor for a nonvisual computing environment. Then we will examine the Audio Rooms metaphor in more detail, identifying matches and mis-matches between the metaphor and the behavior of our proposed environment.

The designer must be convinced that the items which do not match are within the bounds of acceptable behavior. Three types of mis-matches are possible. First, the interface metaphor may contain concepts and actions that are not supported by the computer application. This mis-match will lead to the user trying to perform actions that lead to failure. Second, the computer application may support actions that are not present in the metaphor. These actions will be more difficult for the user to learn and remember. Third, the computer application may behave in ways which directly conflict with what the metaphor predicts. This type of mis-match is the worst since it will cause the user to mistrust the metaphor as a whole. [CMK88]

The second step to designing with metaphors is user testing. No amount of paper designs will predict all the things that users will try to do with a computer application. User testing is an absolute requirement. [CMK88] User testing will undoubtedly uncover mismatches that were not discovered during paper evaluations. There are many other aspects of evaluating a metaphor, for example, determining whether users even like and understand the metaphor. This section has provided a brief overview of designing user interfaces with metaphors. In the following sections, we will examine possible metaphors for a nonvisual computing environment and then we will begin to evaluate our chosen metaphor--Audio Rooms.

4.0 Metaphors for Nonvisual Computing Environments

Throughout most of the history of computers there have been attempts to provide access to computer software and data for those with visual impairments. In the old days of simple textual displays there was an obvious and fairly straightforward method of providing access: text on the screen was sent to a speech synthesis device. This approach worked (and continues to work) acceptably well for those systems and applications which are text based.

In the arena of providing access to graphical systems, however, the attempted solutions have been less successful. Most of these systems try to perform basically the same mapping that was done earlier in the textual systems; that is, take the information which is displayed on the screen and directly map it into some sort of audio and speech output. [BBV91][Fox91][KY87][Van89]

In some systems of this type, visually impaired users use a mouse to manipulate an on-screen cursor. As the cursor passes over objects on the screen the objects speak their names. This approach is relatively straightforward, yet it is naive. One can easily enumerate several problem with this approach:

- Occluded windows cannot be directly accessed
- The mouse is a relative pointing device which gives no indication of the absolute location of the on-screen cursor.
- A visually-impaired user may not notice or be confused by subtle changes to the screen.

metaphor may be more appropriately termed the office metaphor. Although the Macintosh-style graphical environment is based on the desktop metaphor, other graphical environments need not be. One metaphor which is gaining credibility in human--computer interface research laboratories is the so-called rooms metaphor. We shall discuss the rooms metaphor at length later in this chapter.

It should be the goal of such an environment to provide easy access to the basic operating system functionality of a computer while maintaining a level of abstraction which makes the system easy to use. The use of metaphor is commonly employed to make the system intuitive for new users, and to provide common means of using functionality across the entire system.

3.0 Design by Metaphor

We already mentioned the concept of interface metaphors in the previous section. But what exactly is an interface metaphor? How do you use metaphors in designing an interface and how do you evaluate an interface metaphor? Strictly speaking, an interface metaphor is “designed interface actions, procedures, and concepts that exploit specific knowledge that users have of other domains.” [CMK88] The purpose of the metaphor is to give the user instantaneous knowledge about how to interact with the interface. For example, word processing applications exploit the knowledge associated with typewriters by making use of such concepts as margins, spacing, and typing in general. Spreadsheet applications such as Lotus 1-2-3^{tm} are designed to look like common ledger sheets. And desktop computing environments such as the Macintosh Finder contain elements of typical office environments such as filing cabinets, copying machines, and trashcans.

Designing computer interfaces with metaphors is a powerful technique but it is not as simple as it may seem. Interface metaphors allow the user to make assumptions about how a computer application or environment will behave, but you do not want the user to make incorrect assumptions. Once I was working with a colleague to put together a presentation for an upcoming conference. We were using some software on the Macintosh to create our slides. We stored our presentation on our floppy so we could take it with us to the conference. My colleague then dragged the icon for the floppy disk to the Macintosh trashcan. I gasped in horror, wondering why my colleague had suddenly decided to delete all of our hard work. Then the floppy disk was ejected from the disk drive. My college smiled and explained that you could drag a floppy disk to the trashcan to EJECT the disk. I was not amused.

Acknowledging that this example is somewhat trivial, the point is that interface metaphors should be carefully examined before implementation. Getner defines several characteristics of metaphors that can be evaluated: base specificity (how well a metaphor is known), clarity, richness, abstractness, systematically, exhaustiveness, transparency, and scope. [CMK88] Typically, there are at least two steps to designing with metaphors. After a metaphor has been targeted, the designer should enumerate all the possible concepts and actions associated with that metaphor. Then the designer should compare the metaphor’s concepts and actions with the concepts and actions enforced by the computer application.

Unfortunately a portion of the community concerned about access to computers for people who are blind has spent their time hoping that GUIs would go “out of style,” or trying to regulate against the use of GUIs. [Van89] We feel that it would be more beneficial if this community would understand the underlying traits of GUIs that make them powerful, and then demand that interfaces as elegant and powerful as GUIs be made available to them as well.

GUIs are not powerful because they use windows, mice, and icons, per se. Rather it is the underlying benefits of access to multiple information sources, direct manipulation, access to multitasking, and intuitive metaphors which provide the power. The GUI itself is just a single manifestation (perhaps one of many possible manifestations) of an interface which provides these benefits.

2.2 Graphical Computing Environments

In a typical GUI-based computer system there are several components which make up the user’s computing “world.” First, there are the various applications which use graphical user interfaces. These applications are typically visual in nature, and make use of on-screen graphical mechanisms (such as buttons and scrollbars) to control the application.

Second, there is the environment which provides support for performing computer system functions not related to a particular application. Such an environment provides an abstraction for the basic objects in the computer system, such as data files, directories, and so forth, and the basic computer operations, such as copying and deleting. Basically, the environment consists of everything “outside” the applications.

We can describe a computing environment more succinctly by enumerating its functions:

- Provide filesystem and filesystem-related operations, specifically copying, deleting, renaming, and linking.
- Provide mechanisms for starting, stopping, and switching between applications.
- Provide various systems services, such as printing.

It is important to draw this distinction between graphical applications and graphical environments because, while the two may seem superficially (or visually) similar, they provide different functionality. In fact, providing access to graphical applications may have fundamentally different goals from providing access to graphical environments. In this chapter, we will deal primarily with graphical environments.

The Apple Macintosh^{tm} provides a popular graphical environment, Finder^{tm}, which provides an abstraction for executing computer operating system functions. File copying may be accomplished by selecting an icon representing the file and then choosing the “copy” operation from a menu of file operations. File deleting may be accomplished by dragging the icon representing the file into a trashcan icon.

The abstraction employed by the Macintosh is often called a desktop metaphor because the screen layout and supported operations are supposedly reminiscent of a desk. The destination “folder.” Contrast this to a textual interface in which one may accomplish the same task via a command line such as “cp mydoc.tex ~keith/tex/docs.” Of course the syntax for the command line interface may vary widely from system to system.

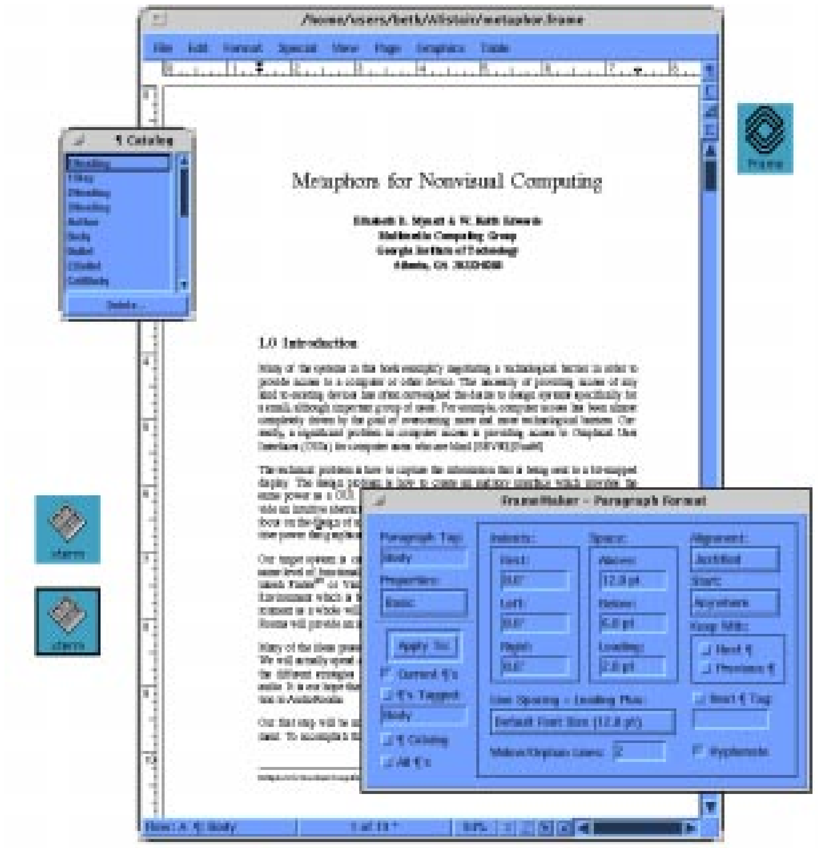
/home/users/beth/vitstairs/metaphor.frame
File Edit Format Special View Page Graphics Table
¶ Catalog
Metaphors for Nonvisual Computing
1.0 Introduction
FrameMaker – Paragraph Format
Frame

FIGURE 1. A typical graphical user interface (GUI)

In addition to direct manipulation, GUIs provide several other important benefits:

- They allow the user to see and work with different pieces of information at one time. Since windows group related information, it is easy for users to lay out their workspaces in a way that provides good access to all needed information.
- An interface to multitasking is easily supported on most GUI-based systems. Each window provides a separate input/output point of control for each process which is running in the system. Processes continue running and users attend to the windows they choose.
- The graphical images used in GUIs lend themselves to the easy implementation of interface metaphors. The graphics support the metaphor by providing a natural mapping between metaphor and on-screen representation of the metaphor.

Next, we will discuss our design methodology, namely the use of metaphors to create an interface that is easy to understand and enjoyable to use. Then we will discuss possible metaphors for a nonvisual computing environment. We will then target our chosen metaphor, Audio Rooms, and evaluate it against real room characteristics. Following the analysis of the Audio Rooms metaphor, we will describe the most powerful strategies that we will use to implement our system. These strategies will revolve around the use of speech and non-speech audio. The remainder of the chapter will provide a brief description of other components in the Mercator environment.

2.0 GUIs and Computing Environments

2.1 The Power of GUIs

For much of their history, computers have traditionally been capable of presenting only textual and numeric data to users. Users reciprocated by specifying commands and data to computers in the form of text and numbers, which were usually typed into a keyboard. This method in which users interacted with their computers was only adequate at best.

More recently, advances in computer power and display screen technology have brought about a revolution in methods of human--computer interaction for a large portion of the user population. The advent of so-called Graphical User Interfaces (or GUIs) has been usually well-received. In this section we shall examine some of the defining characteristics of GUIs, and explore some of the traits that make them useful to the sighted population. This examination will motivate our design of a powerful interface for users with visual impairments.

As implemented today, most GUIs have several characteristics in common:

- The screen is divided into (possibly overlapping) regions called windows. These windows group related information together.
- An on-screen cursor is used to select and manipulate items on the display. This on-screen cursor is controlled by a physical pointing device, usually a mouse.
- Small pictographs, called icons, represent objects in the user's environment which may be manipulated by the user.

Such a system is quite powerful for sighted users for a number of reasons. Perhaps most importantly, there is a direct correlation between the objects and actions which the GUI supports and the user's mental model of what is actually taking place in the computer system. Such a system is often called a direct manipulation interface, since to effect changes in the computer's state, the user simply manipulates the on-screen objects to achieve the desired result. Contrast this to textual interfaces in which there are often arbitrary mappings between commands, command syntax, and actual results. Direct manipulation interfaces are usually intuitive and easy to learn because they provide abstractions which are easy for users to grasp. For example, in a direct manipulation system, users may copy a file by dragging an icon which "looks" like a file to it's

Book chapter to appear in *Extraordinary Human-Computer Interaction,* Alistair Edwards, ed., Cambridge University Press.

Metaphors for Nonvisual Computing

Elizabeth D. Mynatt & W. Keith Edwards
Georgia Institute of Technology

1.0 Introduction

Many of the systems in this book exemplify negotiating a technological barrier in order to provide access to a computer or other device. The necessity of providing access of any kind to existing devices has often outweighed the desire to design systems specifically for a small, although important group of users. For example, computer access has been almost completely driven by the goal of overcoming more and more technological barriers. Currently, a significant problem in computer access is providing access to Graphical User Interfaces (GUIs) for computer users who are blind.[BBV90][Bux86]

The technical problem is capturing the information that is being sent to a bit-mapped display. The design problem is creating a nonvisual interface which provides the same power as a GUI. Modern computing environments generally employ GUIs to provide an intuitive abstraction to operating system concepts and actions. This discussion focuses on the design of a nonvisual computing environment that will provide the same intuitive power that graphical computing environments currently offer.

Our target system is called Audio Rooms. The purpose of this system is to provide the same level of functionality as visually-oriented computing environments such as the Macintosh Finder^{tm} or Visix's Looking Glass^{tm}. Audio Rooms is one part of the Mercator Environment which is being developed at the Georgia Institute of Technology. The environment as a whole will provide access to standard X Window applications while Audio Rooms will provide an intuitive interface for working on a Unix workstation.

Many of the ideas presented in this chapter can be applied to the design of other systems. We will actually spend a great deal of time describing the steps in our design process and the different strategies employed in our design, namely innovative uses of non-speech audio. It is our hope that these ideas can be applied to the design of other systems in addition to Audio Rooms.

Our first step will be an analysis of the requirements for an intuitive computing environment. To accomplish this task, we will examine GUIs and the power they bring to computer interfaces. We will also try to describe the features necessary in a computing environment.