Providing a Structured Method for Integrating Non-Speech Audio into Human-Computer Interfaces

Stephen Anthony Brewster

Submitted for the degree of Doctor of Philosophy

University of York
Human-Computer Interaction Group,
Department of Computer Science

August, *1994*

http://www.dcs.gla.ac.uk/~stephen

Abstract

This thesis provides a framework for integrating non-speech sound into human-computer interfaces. Previously there was no structured way of doing this, it was done in an *ad hoc* manner by individual designers. This led to ineffective uses of sound. In order to add sounds to improve usability two questions must be answered: What sounds should be used and where is it best to use them? With these answers a structured method for adding sound can be created

An investigation of *earcons* as a means of presenting information in sound was undertaken. A series of detailed experiments showed that earcons were effective, especially if musical timbres were used. *Parallel earcons* were also investigated (where two earcons are played simultaneously) and an experiment showed that they could increase sound presentation rates. From these results guidelines were drawn up for designers to use when creating usable earcons. These formed the first half of the structured method for integrating sound into interfaces.

An informal analysis technique was designed to investigate interactions to identify situations where hidden information existed and where non-speech sound could be used to overcome the associated problems. Interactions were considered in terms of events, status and modes to find hidden information. This information was then categorised in terms of the feedback needed to present it. Several examples of the use of the technique were presented. This technique formed the second half of the structured method.

The structured method was evaluated by testing sonically-enhanced scrollbars, buttons and windows. Experimental results showed that sound could improve usability by increasing performance, reducing time to recover from errors and reducing workload. There was also no increased annoyance due to the sound. Thus the structured method for integrating sound into interfaces was shown to be effective when applied to existing interface widgets.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Acknowledgements

I would like to thank my supervisor Alistair Edwards because without his friendship, help, guidance and encouragement this thesis would not have been possible. He introduced me to the area auditory interface design and allowed me the freedom to develop my own ideas. The monster is certainly big enough I just hope it is hairy too!

I would also like to thank Peter Wright. He introduced me to experimental analysis and the psychological aspects of human-computer interaction. He patiently helped me with experimental designs and statistical analysis. He also introduced me to TLX workload measures. Thanks for all the proof reading!

Thanks must go to Robert Stevens for many constructive discussions and for taking my mind off my work. His sense of humour made working together great fun. Thanks also to Ian Pitt for all his help over the years. Many thanks for putting up with all my questions about music and how to use the sound lab!

Thanks to all of my friends in the HCI Group at York who made it such a friendly and stimulating place to work. Thanks also to the members of the University Aikido club for making sure I got at least a few hours of relaxation each week.

The sonically-enhanced buttons described in Chapter 7 benefited from the help of Jon Watte and his knowledge of Macintosh programming.

I would like to thank the trustees of the Gibbs-Plessey Award. This award funded a trip to North America where I met many of the major figures in the field of human-computer interaction and auditory interface design. It also allowed me to present a paper at the first international conference on auditory display. Inspiration for parallel earcons came whilst working with Saul Greenberg at the University of Calgary as part of this award.

Finally, I would like to thank my parents who have always encouraged and supported me during my work on this thesis.

The work described here was funded by SERC studentship 90310837.

Declaration

The literature survey in Chapters 2 and 3 has been published as a Department of Computer Science technical report [31]. The earcon experiments described in Chapter 4 have been published at ICAD’92 and InterCHI’93 [32, 33]. These were joint authored with Peter Wright and Alistair Edwards. The parallel earcon experiment in Chapter 5 has been submitted for publication in the International Journal of Man-Machine Studies [34]. This paper was joint authored with Peter Wright and Alistair Edwards. Referees comments on this paper have been included into the chapter. The description of the event, status and mode analysis technique, its application to the problems of scrollbar kangarooing and loss of position from Chapter 6 and the experimental investigation of the problems in Chapter 7 were published at CHI’94 [35]. This paper was joint authored with Peter Wright and Alistair Edwards. The analysis of the reasons for screen button slip offs discussed in Chapters 6 and 7 was published as a short paper at HCI’94 with Alan Dix [61]. This thesis only exploits those parts of collaborative papers that are directly attributable to the author.

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

The combination of visual and auditory information at the human-computer interface is a natural step forward. In everyday life both senses combine to give complementary information about the world; they are interdependent. The visual system gives us detailed data about a small area of focus whereas the auditory system provides general data from all around, alerting us to things outside our peripheral vision. The combination of these two senses gives much of the information we need about our everyday environment. Dannenberg & Blattner ([23], pp xviii-xix) discuss some of the advantages of using this approach in multimedia/multimodal computer systems:

> "In our interaction with the world around us, we use many senses. Through each sense we interpret the external world using representations and organisations to accommodate that use. The senses enhance each other in various ways, adding synergies or further informational dimensions".

They go on to say:

> "People communicate more effectively through multiple channels. … Music and other sound in film or drama can be used to communicate aspects of the plot or situation that are not verbalised by the actors. Ancient drama used a chorus and musicians to put the action into its proper setting without interfering with the plot. Similarly, non-speech audio messages can communicate to the computer user without interfering with an application".

These advantages can be brought to the multimodal human-computer interface. Whilst directing our visual attention to one task, such as editing a document, we can still monitor the state of other tasks on our machine. Currently, almost all information presented by computers uses the visual sense. This means information can be missed because of visual overload or because the user is not looking in the right place at the right time. A multimodal interface that integrated information output to both senses could capitalise on the interdependence between them and present information in the most efficient and natural way possible. This thesis aims to investigate the creation of such multimodal interfaces.

The classical uses of non-speech sound can be found in the human factors literature (see Deatherage [48] or McCormick & Sanders [116]). Here it is used mainly for alarms and warnings or monitoring and status information. Alarms are signals designed to interrupt the on-going task to indicate something that requires immediate attention. Monitoring sounds provide information about some on-going task. Buxton [38] extends these ideas and suggests that encoded messages could be used to pass more complex information in sound and it is this type of auditory feedback that will be considered here.

The use of sound to convey information in computers is not new. In the early days of computing programmers used to attach speakers to a computer’s bus or program counter [168]. The speaker would click each time the program counter was changed. Programmers would get to know the patterns and rhythms of sound and could recognise what the machine was doing. Another everyday example is the sound of a hard disk. Users often can tell when a save or copy operation has finished by the noise their disk makes. This allows them to do other things whilst waiting for the copy to finish. Sound is therefore an important information provider, giving users information about things in their systems that they cannot see. It is time that sound was specifically designed into computer systems rather than being an add-on or an accident of design that can be taken advantage of by the user. The aim of the research described here is to provide a method to do this.

As DiGiano & Baecker [55] suggest, non-speech audio is becoming a standard feature of most new computer systems. Next Computers [175] have had high quality sound input and output facilities since they were first brought out and Sun Microsystems and Silicon Graphics [154, 185] have both introduced workstations with similar facilities. As Loy [110] says, MIDI interfaces are built in to many machines and are available for most others so that high quality music synthesisers are easily controllable. The hardware is therefore available but, as yet, it is unclear what it should be used for. The hardware manufacturers see it as a selling point but its only real use to date is in games or for electronic musicians. The powerful hardware plays no part in the everyday interactions of ordinary users. Another interesting point is made by DiGiano & Baecker [55]: “The computer industry is moving towards smaller, more portable computers with displays limited by current technology to fewer colours, less pixels, and slower update rates". They suggest that sound can be used to present information that is not available on the portable computer due to lack of display capability.

We have seen that users will take advantage of sounds in their computer systems and that there is sophisticated sound hardware available currently doing nothing. The next step that must be taken is to link these two together. The sound hardware should be put to use to enhance the everyday interactions of users with their computers. This is the area addressed by the research described in this thesis.

1.1.1 Research topics in auditory interface design

In 1989 Buxton, Gaver & Bly [39] suggested six topics that needed further research in the area of auditory interfaces. The areas that they suggest for further investigation partly motivated the work in this thesis. The research topics are:

- *Use of non-speech sound:* Research is needed to find out how people use sound and also to find out about the perception of higher-level musical structures to assess their potential to encode information. What sorts of variations of sounds will prove the most useful and the best associated with a particular meaning? What about the annoyance due to sound?

- *Mapping of information to sound:* Research is needed to explore the mapping of information to sound. Everyday sounds can be mapped to everyday events in a computer. This is intuitive but does not work if there is no everyday equivalent to the operation. Some musical properties map easily into sound (high pitch means up) but are there others? How hard is it to learn new mappings?

- *Sounds in relation to graphics:* How do sounds work in relationship to other types of feedback in the interface? Sounds can complement, replace or work independently of other feedback. Can auditory and visual components be designed to create one coherent system?

- *User manipulation of sounds:* What control should users be given over the parameters of sounds in the interface? Should they be allowed to control volume? What other kinds of controls are needed?

- *Structure of sounds:* Can useful sounds be built-up from smaller components? How are complex structures mapped to sound? How easy is it for listeners to perceive and learn the structures?

- ❖ *System support for sound:* What architectures (hardware and software) are needed to support sound in the interface? What capabilities of sounds are needed? Are MIDI controllers and synthesisers necessary?

These topics are presented so that the description of research in the thesis which follows is put in context. After the contents of the thesis have been described in section 1.6 the work in the thesis will be explained in terms of this research agenda.

One question that might be asked is: Why use sound to present information? A graphical method could be used instead. The drawback with this is that it puts an even greater load on the visual channel. Furthermore, sound has certain advantages. For example, it can be heard from all around, it does not disrupt the user’s visual attention and it can alert the user to changes very effectively. It is for these reasons that this thesis suggests sound should be used to enhance the graphical user interface.

1.2 MOTIVATION FOR RESEARCH INTO AUDITORY INTERFACES

Some of the general advantages that can be gained from adding sound have been described but what are the specific benefits that it offers? There are many reasons why it is important to use sound in user interfaces:

- ❖ To reduce the load on the user’s visual system [114]. Modern, large screen workstations and graphical interfaces use the visual system very intensively. This means that we may miss important information because the visual system is overloaded. Mountford & Gaver ([119], p 322) suggest that the visual display can be overburdened because:

  > “system information is traditionally displayed via graphical feedback that remains on the screen, although it may be obsolete or irrelevant soon after it is shown. The result is often crowded, incomprehensible displays”.

  To stop this overload, information could be displayed in sound. With the limited amount of screen space available, the presentation of some information in sound would allow more important graphical data to be displayed on the screen.

- ❖ Non-intrusive enhancement [103]. Sound can be added to visual displays without interfering with existing tools and skills. If sounds are introduced redundantly with graphics then users will be able to continue to use the systems as before but gain from the advantages of sound. Kramer [103] suggests that the addition of sound will enhance the perceived quality of systems because it allows increased refinement and subtlety.

- The auditory sense is under-utilised. The auditory system is very powerful and would appear to be able to take on the extra capacity. Experiments have shown that a human can distinguish between any two of 400,000 different sounds and remember and identify up to 49 sounds at one time [27]. In certain cases, reaction to auditory stimuli have also been shown to be faster than reactions to visual stimuli [27].

- Sound is attention grabbing [99]. Users can choose not to look at the screen but cannot avoid hearing sound (if they are at the machine). This makes the auditory system very good for presenting alarms and warnings.

- There is psychological evidence to suggest that sharing information across different sensory modalities can actually improve task performance [36, 132] (See Chapter 3 for more on this). Intermodal correlations resulting from sharing between the senses may make the interface more natural. For example, throwing something into the wastebasket and hearing a smashing noise on a computer reflects real life. Sound also has a greater temporal resolution than vision. This means it is good for representing rapidly changing data.

- When information is represented in a visual form users must focus their attention on the output device in order to obtain the presented information and to avoid missing anything. According to Perrott, Sadralobadi, Saberi & Strybel [132] humans view the world through a window of 80° laterally and 60° vertically. Within this visual field focusing capacity is not uniform. The foveal area of the retina (the part with the greatest acuity) subtends an angle of only two degrees around the point of fixation [139]. Sound, on the other hand, is omni-directional. It can be heard without the need to concentrate on an output device, thus providing greater flexibility. Sound does have drawbacks because of its transient nature - once it has been played it cannot be heard again but this may be advantageous, for example, when presenting dynamic, rapidly changing data.

- Some objects or actions within an interface may have a more natural representation in an auditory form. Mountford & Gaver ([119], p 321) suggest sound is useful because “[it] is a familiar and natural medium for conveying information that we use in our everyday lives”. Gaver [74] suggests that sounds are good for providing information on background processes or inner workings without disrupting visual attention. Sound is also a very different medium for representing information than graphics. Bly ([27], p 14) suggests: “… perception of sound is different to visual perception, sound can offer a different intuitive view of the information it presents …". Therefore, sound could allow us to look at information we already have in different ways.

❖ To make computers more usable by visually disabled users. Developments in graphical user interfaces, such as the Apple Macintosh or Microsoft Windows for the PC, have made it harder for blind people to use computers [63]. In older systems, for example PC's running MSDOS, all the information presented was in text. A screen reader could be attached which would read all the text displayed on the screen in synthetic speech. Thus a blind person had access to all of the same information as a sighted person. With the development of graphical displays, information is presented in a pictorial form; users click on a picture of the application they want, instead of reading its name in a list. A screen reader cannot read this kind of graphical information. Providing information in an auditory form could help solve this problem and allow visually disabled persons to use the facilities available on modern computers [121].

Buxton ([38], p 3) claims that sighted users can become so overloaded with visual information that they are effectively visually disabled. He says that if our visual channel is overloaded "we are impaired in our ability to assimilate information through the eyes". Therefore research into displaying information in sound for visually disabled users could be used to help the sighted in these situations. This is also the case in 'eyes-free' interfaces. For instance, where the user must keep visual contact with other elements of the environment or where vision is otherwise impaired, for example in the cockpit of a fighter aircraft.

The area of auditory interfaces is growing as more and more researchers see the possibilities offered by sound because, as Hapeshi & Jones ([89], p 94) suggest, "Multi-media provide an opportunity to combine the relative advantages of visual and auditory presentations in ways that can lead to enhanced learning and recall". There are several examples of systems that use sound and exploit some of its advantages. However, because the research area is still in its infancy, most of these systems have been content to show that adding sound is possible. There are very few examples of systems where sound has been added in a structured way and then formally evaluated to investigate the effects it had. This is one of the aims of this thesis.

1.3 WHAT SOUNDS SHOULD BE USED AND WHERE?

Section 1.2 showed that there are many compelling reasons for using sound at the interface. This brings up two fundamental questions:

❖ What sounds should be used at the interface to communicate information effectively?

❖ Where should sound be used to best effect at the interface?

Prior to the work reported in this thesis there was no structured method a designer could use to add sound. It had to be done in an *ad hoc* manner for each interface. This led to systems where sound was used but gave no benefit, either because the sounds themselves were inappropriate or because they were used in inappropriate places. If sounds do not provide any advantages then there is little point in the user using them. They may even become an annoyance that the user will want to turn off. However, if the sounds provide information users need then they will not be turned off. The work described in this thesis answers these two questions and from the answers provides a structured method to allow a designer (not necessarily skilled in sound design) to add effective auditory feedback that will improve usability. The structured method provides a series of steps that the designer can follow to find out where to use sound and then to create the sounds needed.

There are several different methods for presenting information in sound and two of the main ones are: *Auditory icons* [74] and *earcons* [25]. Auditory icons use natural, everyday sounds to represent actions and objects within an interface. The sounds have an intuitive link to the thing they represent. For example, selecting an icon might make a tapping sound because the user presses on the icon with the cursor. Auditory icons have been used in several interfaces. Whilst they have been shown to improve usability [79] no formal evaluation has taken place. One drawback is that some situations in a user interface have no everyday equivalents and so there are no natural sounds that can be used. For example, there is no everyday equivalent to a database search so a sound with an intuitive link could not be found.

Earcons are the other main method of presenting information in sound. They differ from auditory icons in the types of sounds they use. Earcons are abstract, synthetic tones that can be used in structured combinations to create sound messages to represent parts of an interface. Earcons are composed of motives, which are small sub-units that can be combined in different ways. They have no intuitive link to what is represented; it must be learned. Prior to the research described in this thesis, earcons had never been evaluated. The best ways to create them were not known. It was not even clear if users would be able to learn the structure of earcons or the mapping between the earcon and its meaning. This lack of knowledge motivated the investigation of earcons carried out in this thesis. When more was known about earcons a set of guidelines for their production could be created. These guidelines should also embody knowledge about the perception of sound so that a designer with no skill in sound design could create effective earcons.

Neither of the two sound presentation methods above give any precise rules as to where in the interface the sounds should be used. The work on auditory icons proposed that they should be used in ways suggested by the natural environment. As discussed above, this can be a problem due to the abstract nature of computer systems; there may be no everyday equivalent of the interaction to which sound must be added. This work also only uses sounds redundantly with graphical feedback. Sounds can do more than simply indicate errors or supply redundant feedback for what is already available on the graphical display. They should be used to present information that is not currently displayed (give more information) or present existing information in a more effective way so that users can deal with it more efficiently. A method is needed to find situations in the interface where sound might be useful and this thesis presents such a method. It should provide for a clear, consistent and effective use of non-speech audio across the interface. Designers will then have a technique for identifying where sound would be useful and for using it in a more structured way rather than it just being an *ad hoc* decision.

In the research described in this thesis sound is used to make explicit information that is hidden in the interface. Hidden information is an important source of errors because often users cannot operate the interface effectively if information is hidden. There are many reasons why it might be hidden: It may not be available because of hardware limitations such as CPU power; it may be available but just difficult to get at; there may be too much information so that some is missed because of overload; or the small area of focus of the human visual system may mean that things are not be seen. This thesis describes an informal analysis technique that can be used to find hidden information that can cause errors. This technique models an interaction in terms of *event*, *status* and *mode* information and then categorises this in terms of the feedback needed to present it.

Many uses of sound at the human-computer interface are never evaluated. One reason for this is that research into the area is very new so that example systems are few in number. Most of the interfaces developed just aimed to show that adding sound was possible. However, for the research area to develop and grow it must be shown that sound can effectively improve usability. Therefore, formal testing of sonically-enhanced interfaces is needed. One aim of this research is to make sure that the effects of sound are fully investigated to discover its impact. In particular annoyance is considered. This is often cited as one of the main reasons for not using sound at the interface. This research investigates if sound is annoying for the primary user of the computer system.

The answers to the two questions of where and what sounds are combined to produce a structured method for adding sound to user interfaces. The analysis technique is used to find where to add sound and then the earcon guidelines are used to create the sounds needed. This method is tested to make sure the guidelines for creating sounds are effective, the areas in which to add sound suggested by the analysis technique work and that usability is improved.

1.4 A DEFINITION OF TERMS

1.4.1 Usability

In the section above one of the aims of the thesis was shown to be creating a structured method for adding sound that would increase usability. What is meant by usability in this case? In ISO standard 9241-11 (reported in [19], p 135 and also described in [126]) it is defined as: “The effectiveness, efficiency and satisfaction with which specific users achieve specified goals in particular environments”. Bevan & Macleod [19] suggest that effectiveness can be measured by accuracy, efficiency by time and satisfaction by subjective workload measures. This definition of usability will be used when measuring the effectiveness of the structured method for adding sound.

1.4.2 Multimedia and multimodal systems

The research described in this thesis aims to create multimodal interfaces. What is a multimodal interface and how does it differ from a multimedia one? There are, as yet, no accepted definitions of the terms multimedia and multimodal as Alty and Mayes both describe [3, 115]. This thesis uses the definitions proposed by Mayes:

❖ *Multimedia*: A *medium* is a carrier of information, for example printed paper, video or a bit-mapped display. As Mayes says (p 2): “It may be used to refer to the nature of the communication technology”. A multimedia computer system is one that is capable of the input or output of more than one medium [22]. In this definition a computer screen is a multimedia device because it can display text, graphical images and video. The medium of the display can contain pictures, text, etc.

❖ *Multimodal*: The term *mode* has many meanings. In computer system dialogues modes put an interpretation on information and affect what the user is able to do at any given point in the system (see Chapter 6 for more on this). Mode refers to the state of the system. Mode can also refer to the human sense that is used to perceive the information - the *sensory modality* (see Chapter 2 for more on this). This is the standard psychological definition. In this thesis a multimodal interface is defined as one that presents information in different sensory modalities, specifically visual and auditory.

Almost all computer systems are multimedia by this definition. They all have the ability to present information via different media such as graphics, text, video and sound. They are not all multimodal however. Most of the different media they use present information to the visual system. Very few systems make much of their capacity to produce sound. Errors are sometimes indicated by beeps but almost all interactions take place in the visual modality. The aim of this research is to broaden this and make everyday interactions with computers use the auditory modality as well as the visual.

1.4.3 Musical notation used in the thesis

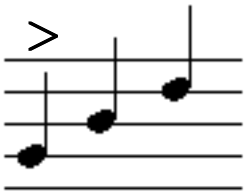
Standard musical notation is used to describe the earcons in this thesis. In this very brief description only the parts of musical notation used by the sounds in the thesis are described. For a more detailed description of the notation used see Scholes [148]. The earcons used are based around the quarter note. Whole notes are four times as long as quarter notes, half notes twice as long, eighth notes half the length, etc. A quarter note rest is a period of silence for the length of a quarter note. These time divisions and their iconic notations are:

w= Whole note   h= Half note   q= Quarter note

e = Eighth note   x = Sixteenth note   ˛= Quarter note rest

= Eighth note rest

The arrangement of notes on the stave (the series of horizontal lines) defines the rhythm of the earcon. An example earcon might look like this:



These are three quarter notes of increasing pitch. A note with a '>' above it is accented (played slightly louder than normal), with a '<' it is muted. A sequence of notes with a '<' underneath indicates that they get louder (crescendo) and with a '>' they get quieter (decrescendo). The height of the note on the stave indicates its relative pitch. This is only a very simple overview of musical notation.

1.4.4 Pitch notation used in the thesis

In addition to describing the notes and rhythms used the octave of the notes must be specified. There are eight octaves of seven notes in the western diatonic system [148]. There are many different systems for notating pitch. The one used in this thesis is described in Scholes. In this commonly used system a note, for example ‘C’, is followed by an octave number, for example:

		Middle C		
C_1	C_2	C_3	C_4	C_5
1046 Hz	523 Hz	261 Hz	130 Hz	65 Hz

So A above middle C (440 Hz) would be $A_{3.}$ This system will be used throughout the thesis to express pitch values.

1.5 THESIS AIMS

In this section the main aims of the thesis will be summarised. The overall aim of this research is to provide a structured method that designers can use to integrate sounds into human-computer interfaces. By doing this it is also hoped that sound will be shown to be effective at communicating information and able to increase the usability of systems. Before the method can be created two questions must be answered:

What sounds should be used at the human-computer interface? The main aims of this part of the work are:

- To investigate whether earcons are an effective method for presenting structured information in sound;
- To show the best way to construct earcons;
- To investigate whether their rate of presentation could be increased so that they can keep pace with interactions;
- To improve upon the current rules for creating them and produce a set of guidelines for interface designers.

Where should sound be used at the human-computer interface? The main aims of this part of the work are:

- To analyse some interactions to investigate whether there are problems due to hidden information;

- ❖ To find out if using event, status and mode analysis will make useful predictions about where to use sound;
- ❖ To see if the feedback to make the hidden information explicit can be modelled;
- ❖ To produce an analysis technique that an interface designer could use to find where to add effective sounds.

These two components will be brought together and the structured method will be evaluated. The aim of the evaluation will be:

- ❖ To determine the effectiveness of the structured method by investigating if the sounds added improve usability;
- ❖ To find out if sounds used in this way are annoying to the primary user of the computer system.

1.6 CONTENTS OF THE THESIS

Figure 1.1 shows the structure of the thesis and how the chapters contribute to the two questions being investigated. Chapters 2 and 3 set the work in context, Chapters 4 and 5 investigate what sounds are the best to use, Chapter 6 shows where sound should be used and Chapter 7 brings all the work together to show the structured method in action. The following paragraphs give an overview of each chapter.

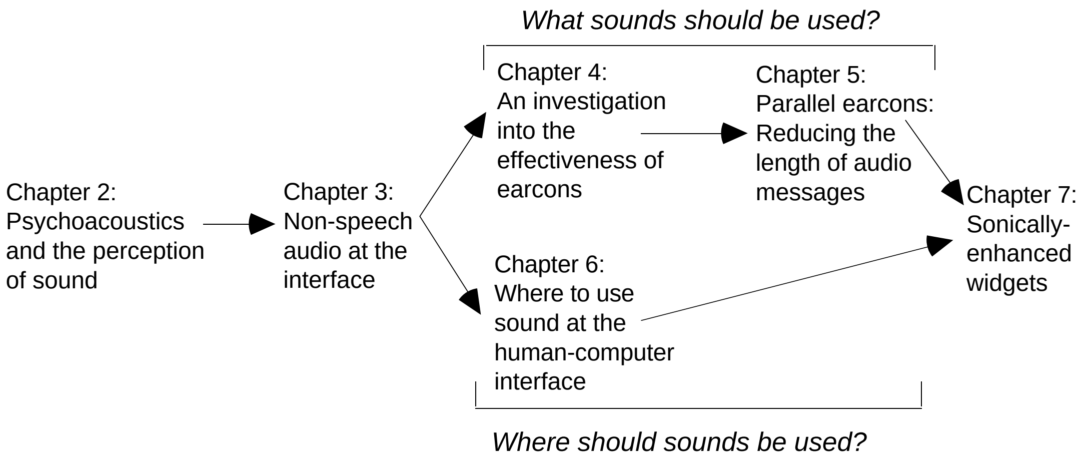

What sounds should be used?
Chapter 2: Psychoacoustics and the perception of sound
Chapter 3: Non-speech audio at the interface
Chapter 4: An investigation into the effectiveness of earcons
Chapter 5: Parallel earcons: Reducing the length of audio messages
Chapter 6: Where to use sound at the human-computer interface
Chapter 7: Sonically-enhanced widgets
Where should sounds be used?

Figure 1.1*: Structure of the thesis.*

Chapter 2 gives an introduction to *psychoacoustics*, the study of the perception of sound. This is important when designing auditory interfaces because using sounds without regard for psychoacoustics may lead to the user being unable to differentiate one sound from another or being unable to hear the sounds. The main aspects of the area are dealt with including: Pitch and loudness perception, timbre, localisation and auditory pattern recognition. The chapter concludes by suggesting that a set of guidelines incorporating this information would be useful so that auditory interface designers would not need have an in-depth knowledge of psychoacoustics.

Chapter 3 provides a background of existing research in the area of non-speech audio at the interface. It gives the psychological basis for why sound could be advantageously employed at the interface. It then goes on to give detailed information about the main systems that have used sound including: Soundtrack, auditory icons, earcons and auditory windows. The chapter highlights the fact that there are no effective methods in existence that enable a designer to find where to add sound to an interface. It also shows that none of the systems give any real guidance about designing the types of sounds that should be used. One of the main systems, earcons, has not even been investigated to find out if it is effective.

Chapter 4 describes a detailed experimental evaluation of earcons to see whether they are an effective means of communication. An initial experiment shows that earcons are better than unstructured bursts of sound and that musical timbres are more effective than simple tones. The performance of non-musicians is shown to be equal to that of trained musicians if musical timbres are used. A second experiment is then described which corrects some of the weaknesses in the pitches and rhythms used in the first experiment to give a significant improvement in recognition. These experiments formally show that earcons are an effective method for communicating complex information in sound. From the results some guidelines are drawn up for designers to use when creating earcons. These form the first half of the structured method for integrating sound into user interfaces.

Chapter 5 extends the work on earcons from Chapter 4. It describes a method for presenting earcons in parallel so that they take less time to play and can better keep pace with interactions in a human-computer interface. The two component parts of a compound earcon are played in parallel so that the time taken is only that of a single part. An experiment is conducted to test the recall and recognition of parallel compound earcons as compared to serial compound earcons. Results show that there are no differences in the rates of recognition between the two types. Non-musicians are again shown to be equal in performance to musicians. Parallel earcons are shown to be an effective means of increasing the presentation rates of audio messages without compromising recognition. Some extensions to the earcon creation guidelines of the previous chapter are proposed.

Chapter 6 investigates the question of where to use sound. It describes an informal analysis technique that can be applied to an interaction to find where hidden information may exist and where non-speech sound might be used to overcome the associated problems. Information may be hidden for reasons such as: It is not available in the interface, it is hard to get at or there is too much information so it cannot all be taken in. When information is hidden errors can occur because the user may not know enough to operate the system effectively. Therefore, the way this thesis suggests adding sound it to make this information explicit. To do this, interactions are modelled in terms of events, status and modes. When this has been done the information is categorised in terms of the feedback needed to present it. Four dimensions of feedback are used: Demanding versus avoidable, action-dependent versus action-independent, transient versus sustained, and static versus dynamic. This categorisation provides a set of predictions about the type of auditory feedback needed to make the hidden information explicit. In the rest of the chapter detailed analyses of many interface widgets are shown. This analysis technique, with the earcon guidelines, forms the structured method for integrating sound into user interfaces.

Chapter 7 demonstrates the structured method in action. Three sonically-enhanced widgets are designed and tested based on the method. The chapter discusses problems of annoyance due to sound and some ways it may be avoided. The first experiment tests a sonically-enhanced scrollbar. The results show that sound decreases mental workload, reduces the time to recover from errors and reduces the overall time taken in one task. Subjects also prefer the new scrollbar to the standard one. Sonically-enhanced buttons are tested next. They are also strongly preferred by the subjects and they also reduce the time taken to recover from errors. Finally, sonically-enhanced windows are tested. Due to a problem with the experiment it is not possible to say whether they improve usability. In all of the three experiments subjects did not find the sounds annoying. The structured method for adding sound is therefore shown to be effective.

Chapter 8 summarises the contributions of the thesis, discusses its limitations and suggests some areas for further work.

1.6.1 The thesis in terms of the research topics in auditory interface design

How does the work in this thesis fit into the research agenda described in section 1.1.1? The investigation of earcons in Chapters 4 and 5 falls into three of these areas. It investigates the *use of non-speech sound*. The experiments investigate the best types of sounds to use; the best timbres, pitches, rhythms, etc. The work deals with the *mapping of information to sound* and how hard these mappings are to learn. Finally, the chapter looks at the *structure of sounds*. Earcons are investigated to find out if listeners can extract and learn their structure.

Chapter 6 investigates *mapping information to sound*. The agenda suggests that a method for translating events and data into sound is needed and this is what the research provides. It gives an analysis technique that models hidden information and from this produces rules for creating sounds. The chapter also investigates the *sound in relation to graphics*, suggesting that sound and graphics can be combined to create a coherent system.

Chapter 7 again looks at the *use of non-speech sound* and particularly at the annoyance due to sound. It considers sound in relation to graphical feedback. Sounds are shown complementing and replacing graphics.

The thesis does not investigate system support for sound, although from the research the types of sounds necessary in an interface are shown. This knowledge could then be used when deciding what hardware and software are needed to support sound in a computer system. The research also does not investigate user manipulation of sounds.

The work undertaken for this thesis has been shown to address many of the major research issues that Buxton *et al.* suggest are important for the future of auditory interfaces. The answers gained from this thesis will extend knowledge of how sounds can be used at the interface.

Chapter 2: Psychoacoustics and the Perception of Sound

2.1 INTRODUCTION

The auditory interface designer must be conscious of the effects of psychoacoustics, the study of the perception of sound, when designing sounds for the interface. Frysinger ([72], p 31) says:

> "The characterisation of human hearing is essential to auditory data representation because it defines the limits within which auditory display designs must operate if they are to be effective".

Using sounds without regard for psychoacoustics may lead to the user being unable to differentiate one sound from another, unable to hear the sounds or unable to remember them. Subsequent chapters of this thesis put forward a set of guidelines for the use of sounds in human-computer interfaces. To do this, a detailed knowledge of the perceptual issues involved is needed. It is to provide a strong basis from which to create the rest of the sounds used in the thesis that a detailed description of psychoacoustics follows.

The fundamental topics of psychoacoustics important for multimodal interface design are described and implications for the design of interfaces given. To begin this chapter a brief summary of sound will be given. This is followed by detailed sections on the main components of sound and factors affecting their perception. Localisation, or the ability to position a sound source in space, is then discussed. Auditory pattern recognition then follows which is an important topic if multiple sounds are to be played simultaneously in an interface. A summary of this chapter is provided in [31].

2.1.1 The components of sound

What is sound ? Sounds are pressure variations that propagate in an elastic medium (in this case, the air). A sound is made up from three basic components: Pitch, timbre and intensity. These can be defined as follows:-

- *Pitch* is related to the frequency of the tone. Pitch may be defined as the attribute of auditory sensation in terms of which sounds may be ordered on a musical scale [6]. In the western musical system there are 96 different pitches arranged into 8 octaves of 12 notes. Tones separated by an octave have the frequency ratio 2:1. For example, middle C is 261.63Hz, the octave above this is at 523.25Hz and the octave below at 130.81Hz.

- *Loudness* is the perceived intensity of a sound. This is defined by the amplitude of the sound wave.

- *Timbre* is the ‘quality’ of the sound. It may be defined as the attribute of auditory sensation in terms of which a user can judge two sounds similarly presented and having the same loudness and pitch are dissimilar [5]. Timbre is that which allows a listener to distinguish between a piano and a violin playing the same note. The timbre is defined by the shape of the waveform; a sine wave is a ‘pure’ tone. Definitions of timbre are often quite vague which leads Bly ([27], p 29) to describe timbre as “the multidimensional wastebasket category for everything that is not pitch nor loudness”.

These components will be described in greater detail below. In the study of psychoacoustics two types of sounds are used. Simple tones (also called pure tones) are made up from just one sine wave (see Figure 2.1). They are used as they elicit a simple response from the auditory system that can be easily measured (although this response will not be similar to that gained from listening to natural sounds).

Complex (harmonic) tones are made up of many sine waves and are similar to the sounds normally heard in the natural environment. The lowest sine wave component defines the pitch of the tone and is called the *Fundamental Frequency* (F_0). There are then any number of harmonic frequencies (F_1 - F_n) above this which are integral multiples of the fundamental (see Figure 2.2). In the figure, the top wave is the fundamental frequency, where F_0 = 100 Hz. The wave below is the first harmonic with a frequency half of the F_0 (F_1 = 200 Hz), the one below that has half the frequency of the first harmonic (F_2 = 400 Hz). The final, complex wave is made by adding the three simple ones together. Another example might be a complex tone with a fundamental of 200 Hz, then the harmonics would be F_1 = 400 Hz, F_2 = 600 Hz, F_3 = 800 Hz, etc. Rasch & Plomp [138] showed that a listener can differentiate the first five to seven harmonics in a complex tone. Multiple harmonics make complex tones easier to recognise as the ear has more information to work with.

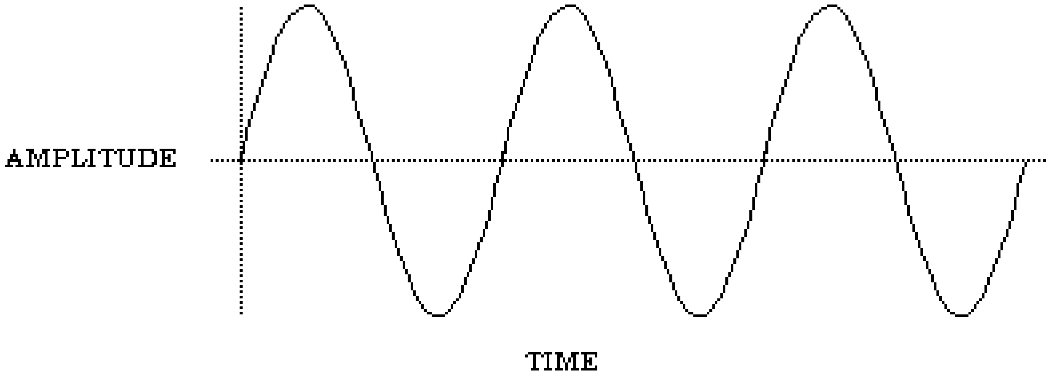


Figure 2.1*: A simple tone.*

One other important factor is the temporal resolution of sound. The minimum temporal resolution of sound is 2 msec, for touch it is 10 msec and for vision it is 100 msec [70]. This shows that two auditory stimuli can be heard as separate when presented with a gap of 2 msec between them. This is a much smaller difference than with two visual stimuli. This has an important impact on the use of sound in user interfaces. Data could be presented in sound at much greater rates than graphically. Therefore, if information must be presented rapidly sound may be a more effective medium than graphics.

2.2 PITCH PERCEPTION

Pitch is one of the most useful and easily controlled components of sound. First the perception of pitch will be described to give background and then some factors affecting pitch perception will be discussed. The way that pitch is perceived is, as yet, not fully understood. There are two main theories which attempt to explain it: Place theory and temporal theory.

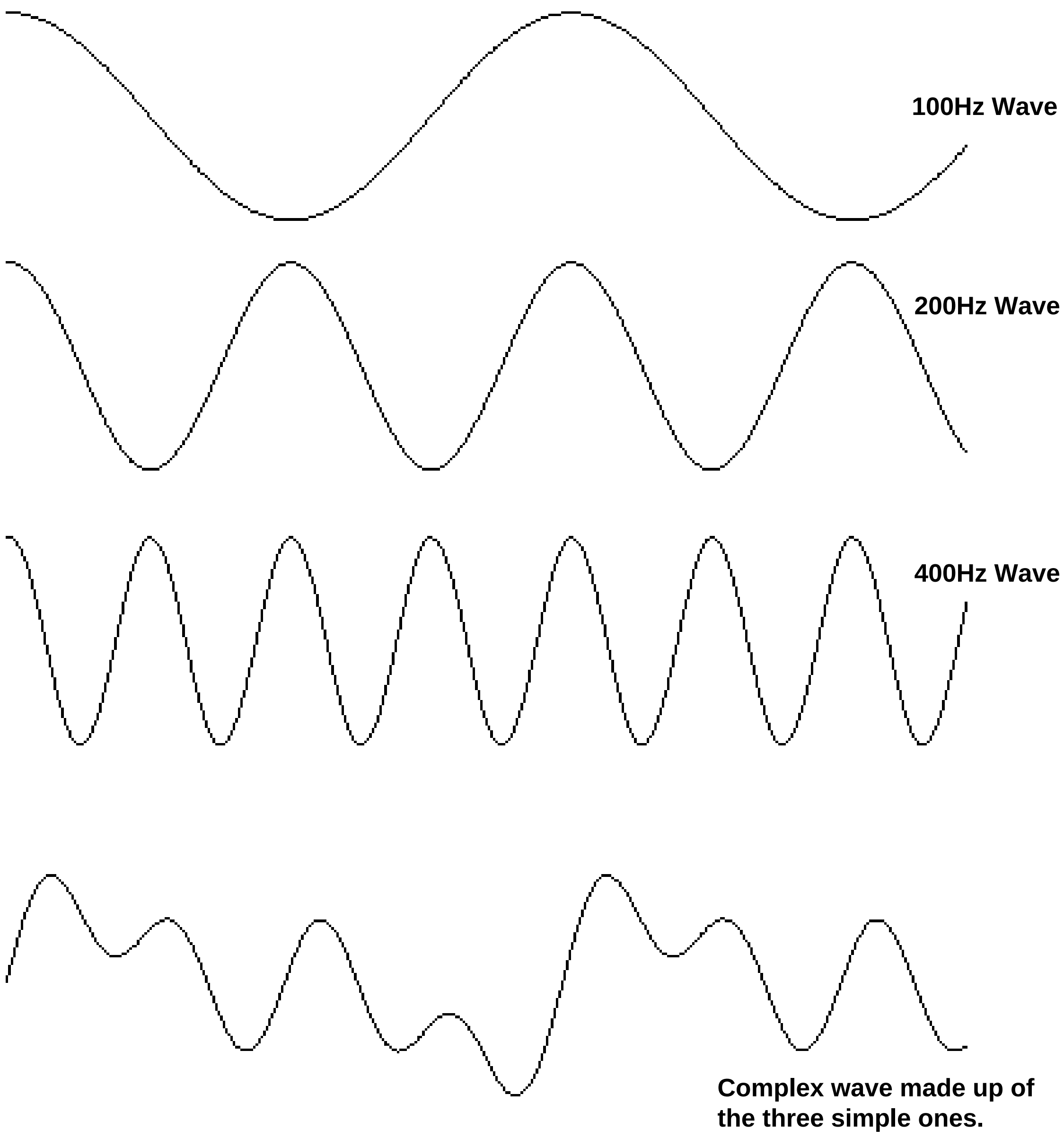


Figure 2.2: *A set of sine waves from a complex tone. The first three add together to produce the fourth, complex tone.*

2.2.1 Place Theory

This is the classical explanation of pitch perception, developed by Von Békésy (1947, from Moore [118]). It suggests that sound undergoes spectral analysis within the inner ear so that different frequencies excite different places along the Basilar Membrane (this membrane vibrates in response to a sound and causes nerve impulses to be transmitted to the brain. For more detail on this see Moore ([118], pp 17-28)). For a pure tone the pattern of excitation along the membrane will be limited to the position corresponding to the frequency. For a complex tone there will be a pattern of excitation corresponding to the fundamental and all the harmonics. A problem with place theory is that excitation pattern along the Basilar membrane may not be at a maximum at the position corresponding to the fundamental, but the correct pitch will still be heard. Another problem occurs with tones of 400 Hz or below, as the sound wave stimulates the entire Basilar membrane equally [83] but humans can normally hear pitches down to approximately 20Hz so other processes must be involved.

2.2.2 Temporal Theory

Temporal theory is based on the ideas of place theory but time plays a much more important role. The perceived pitch of a tone is related to the time pattern of neural impulses the tone evokes. Nerve firings tend to occur at a particular phase in the stimulating waveform (this is called Phase Locking). For example, a 500Hz tone will have a period of 2ms, so that intervals between nerve firings might be 2ms, or 4ms, or 6ms etc. Thus, the intervals between successive firings approximate integral multiples (i.e. the harmonics) of the stimulus. For more information see Moore ([118] Chapter 5). Auditory nerves cannot fire more than about 1000 times a second [83] so this theory would be limited to tones below 1000Hz. Therefore temporal theory cannot fully explain pitch perception and it is probably the case that a combination of both place and temporal theories work to control the perception of pitch.

2.2.3 The problem of the missing fundamental

One problem that the theories cannot explain is that of the *missing fundamental*. A complex tone is presented to the ear with, for example, a F_0 = 200Hz and harmonics of 400Hz, 600Hz, 800Hz, etc. If the fundamental frequency is then filtered out *the perceived pitch of the sound does not change.* There is no sound energy to stimulate the Basilar membrane at the point corresponding to the fundamental, but the same pitch is heard (although the timbre may change slightly). The auditory system does not take the first harmonic and let it become the fundamental, which would raise the pitch, because the harmonic structure would be wrong. A tone of F_0 = 400Hz would have harmonics of 800Hz, 1200Hz, 1600Hz etc., not 600Hz, 800Hz, etc. (that the F_0 = 200Hz tone had). The auditory system seems to calculate what the fundamental should be from the harmonics. This was first discovered by Schouten (1940, from Scharf & Houtsma [147]). He found that all but a few mid-range frequencies, around 1800Hz to 2200Hz, could be filtered out and the same pitch would still be heard, although with a different timbre. All this implies that the perception of pitch does not require stimulation of the Basilar membrane at a point which would correspond to the fundamental. Gelfand ([80] pp 284-5) gives further explanation of this. According to Frysinger [72] this effect could be useful in auditory displays if masking tones (see the next section on loudness perception) were present at around the same frequency as the fundamental because the perception of the pitch would not be affected. This is another advantage of using complex tones for interface sonification.

2.2.4 Factors affecting pitch perception

Now the mechanisms for the perception of pitch have been explained, some factors which affect the perception will be discussed. Three main factors affect the perceived pitch of a tone: Frequency, intensity and timbre (frequency being the most important). The relationship between perceived pitch and frequency is non-linear. Tripling the frequency from 1kHz to 3kHz only doubles the perceived pitch (see Figure 2.3). Pitch is almost a logarithmic function of frequency [39]. On the graph pitch is measured in *Mels*, a subjective pitch scale. A tone of 1000Hz (40dB SPL) has a defined pitch of 1000 Mels. The pitch in Mels of other tones with different frequencies are then related to this. A sound with a pitch subjectively twice that of a 1000Hz tone would be 2000 Mels; a tone with 'half pitch' would be 500 Mels [138]. This is important for the creation of auditory interfaces. If, for example, one wanted to create a set of tones equally spaced in pitch then simply making them each the same frequency apart would not be enough because the relationship between pitch and frequency is logarithmic.

Intensity also affects perceived pitch. At less than 2kHz an increase in intensity increases the perceived pitch. At 3kHz and over an increase in intensity decreases the perceived pitch [80]. Perceived pitch is also affected by timbre. Bright sounds, i.e. those

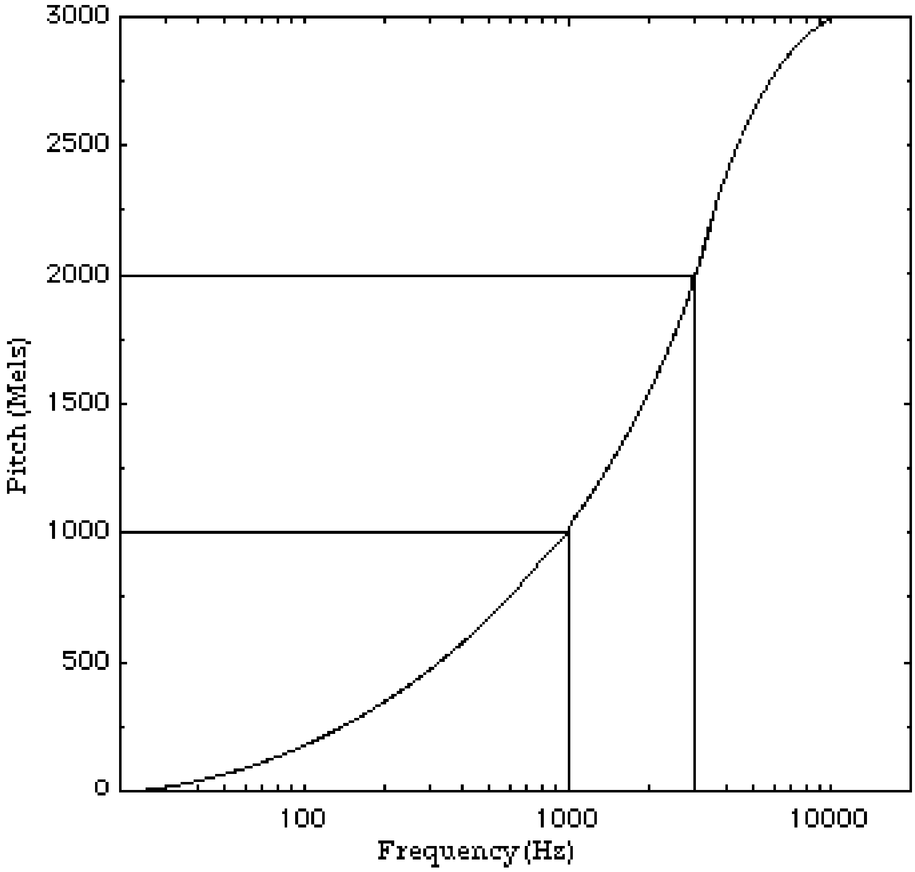


***Figure 2.3:** The relationship between pitch and frequency (from* [80]*).*

with a relatively greater amount of high-frequency energy, sound higher pitched than dull ones [39]. This would be important, for example, if two tones of the same frequency but different timbre were to be compared. Their pitches might be heard as different where it would be expected they would sound the same. As Buxton, Gaver & Bly say (p 2.10): “It is important to be aware of the myriad interactions between pitch and other attributes of sound when using pitch…”.

Humans are able to distinguish very small changes in levels of pitch. The smallest detectable change is called the frequency difference limen. Gelfand [80] showed that this becomes larger as frequency increases and smaller as frequency decreases. At low frequencies, around 200Hz and less, differences of less than 1Hz can be detected. The frequency difference limen is approximately 16Hz at 4kHz and 68Hz at 8kHz (these figures relate to sounds at 40dB SPL, changing the intensity will change the frequency difference limen). The implications of this are important when creating auditory interfaces. If tones with different pitches are to be presented care must be taken so that the pitch differences are larger than the frequency difference limen at that frequency, otherwise they will not be distinguishable. Another point to bear in mind is that the maximum and minimum discernible frequency range decreases with age; for most adults the threshold of hearing rises rapidly above 15kHz. Thus for an adult to hear a high-frequency tone a much higher sound intensity is needed (see below for more on intensity). At their peak, children can hear pitches from 20Hz to 20kHz. As Sumikawa [164] reports, by the age of 30 an average person can hear frequencies no higher than 18kHz, by 50 the limit is 14kHz and by 70 is 10kHz. For auditory interfaces to be usable by the general population, Patterson suggests frequency levels should be kept to below 5kHz and above 130Hz [128]. This is also important for hardware reasons as well. If neither extreme of pitch is used then sounds will be reproducible even on poor quality hardware.

Although listeners are good at detecting small differences in pitch they are not good at making absolute pitch judgements, unless they have *perfect pitch* which is rare [39]. Moore [118] suggests that only 1% of the population have this ability. Therefore, if absolute judgements are required in an interface that uses sound, pitch may not be the best method of presenting the information. Another important factor is ‘tone deafness’. Moore (p 147) says that this is a misnomer: “…nearly everyone is able to judge that two tones are different in pitch when their frequency difference exceeds a certain amount”. One common problem is a listener being unable to assign a direction to a pitch change; he/she can hear the tones are different but cannot say which one is higher in pitch. Moore says that very often this can be overcome by practice. This is again very important to be aware of when designing an auditory interface.

Pitch is not a single perceptual parameter [39]. In addition to *pitch height* discussed above, there is a circular component to pitch (often called the *circle of fifths*). Sounds an octave apart sound very similar, in some cases more similar than pitches closer in frequency [52]. Blattner *et al.* [25] addressed this problem in their work on presenting information in sound. They suggest that the notes used in auditory stimuli should be from the same octave and in the same scale to avoid problems of octave perception (see the section on earcons in the next chapter for more details).

Mansur, Blattner & Joy ([113], p 171) give evidence of one other important psychoacoustic effect. They report that:

> “There appears to be a natural tendency, even in infants, to perceive a pitch that is higher in frequency to be coming from a source that is vertically higher in space when compared to some lower tone.”

This phenomenon is obviously a very important one when creating an auditory interface as it could be used to give objects a spatial position. If only stereo position is available to provide spatial cues in the horizontal plane (see below) then pitch could provide them in the vertical plane.

2.3 LOUDNESS PERCEPTION

Loudness is the perceived volume of a sound and is related to intensity. Intensity is the physical volume of the sound as defined by the amplitude of the sound wave. The amplitude of the sound wave entering the ear causes a certain amount of stimulation of the Basilar membrane (at a position defined by the pitch of the tone). The greater the amplitude, the greater the stimulation and hence the greater the perceived loudness. Perceived loudness is usually measured in *Sones*. One sone is the loudness of 1000Hz tone at 40dB SPL. Doubling the sone level doubles the perceived loudness of a tone [80]. The physical intensity is normally measured in decibels. This is the logarithmic ratio between 2 levels, one of which is a reference level. A standard reference level often used is the Sound Pressure Level (SPL) which is set at 20µPa ([118] pp 8-9). This corresponds to the normal threshold of hearing. The ear has an enormous sensitivity range; the most intense sound a human can hear has a level 120dB greater than the faintest sound. This corresponds to a ratio of 1,000,000,000,000:1 [118]. Some example decibel values are: 20dB: a quiet country area at night; 70dB: normal conversation level; and 120dB: the maximum that can be endured without damage to the auditory system. Absolute threshold is the minimum detectable sound level; at 1kHz the threshold for a person with normal hearing is 6.5dB.

2.3.1 Factors affecting loudness perception

The main factors that effect perceived loudness are: Intensity, frequency and masking.

Intensity

Figure 2.4 shows physical intensity (dB SPL) plotted against perceived loudness (Sones). It can be seen that loudness is proportional to intensity. A 10dB increase in intensity doubles the perceived loudness. For example, an increase from 40dB to 50dB causes loudness to increase from 1 to 2 sones. Near threshold levels small increases in intensity result in larger increases in loudness.

Frequency

Loudness is also affected by frequency. This effect can be heard when adjusting the bass and treble controls on a stereo amplifier; by emphasising the higher or lower frequency parts of the sound the loudness appears to change, although the volume control has not been touched.

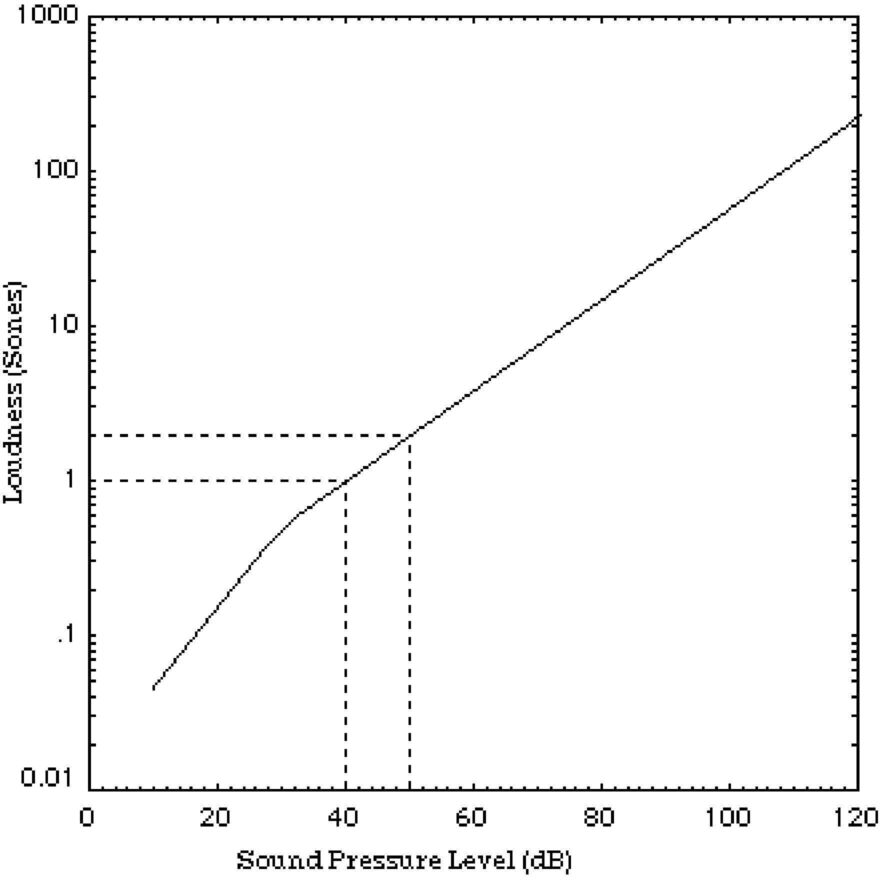


Figure 2.4: *The relationship between sound intensity and loudness (from* [147]*).*

Figure 2.5 shows some equal loudness contours: Anywhere along the contours sounds of different frequencies have the same loudness [80]. Each contour is based on the loudness of a 1000Hz tone at a given intensity. The contours are measured in *Phons* – a measure of loudness level. The loudness level of a sound is the level of the 1000Hz tone to which it sounds as loud. Therefore, the contours show the intensity needed for a given frequency to make it sound as loud as a 1000Hz tone of given intensity. For example, for a 100Hz tone to sound as loud as 1000Hz tone at 20dB, it must be 40dB; For a 5000Hz tone to sound as loud it only needs to be 16dB. The graph also shows that for very low (20Hz to 100Hz) and very high (7kHz to 10kHz) frequency sounds much higher intensity levels are needed to make them equally as loud as mid-range frequencies. The contour labelled ‘MAF’ on the graph shows the Minimum Audible Field, i.e. the threshold level of hearing at various frequencies.

These two factors show that care must be taken when creating sounds for auditory interfaces which are to be of differing frequencies but the same loudness. The intensity of the sounds may need to be adjusted with the changing frequencies to create the perceptual effect of equal loudness. As mentioned above for pitch, the interactions between loudness and the other attributes of sound are many so care must be taken.

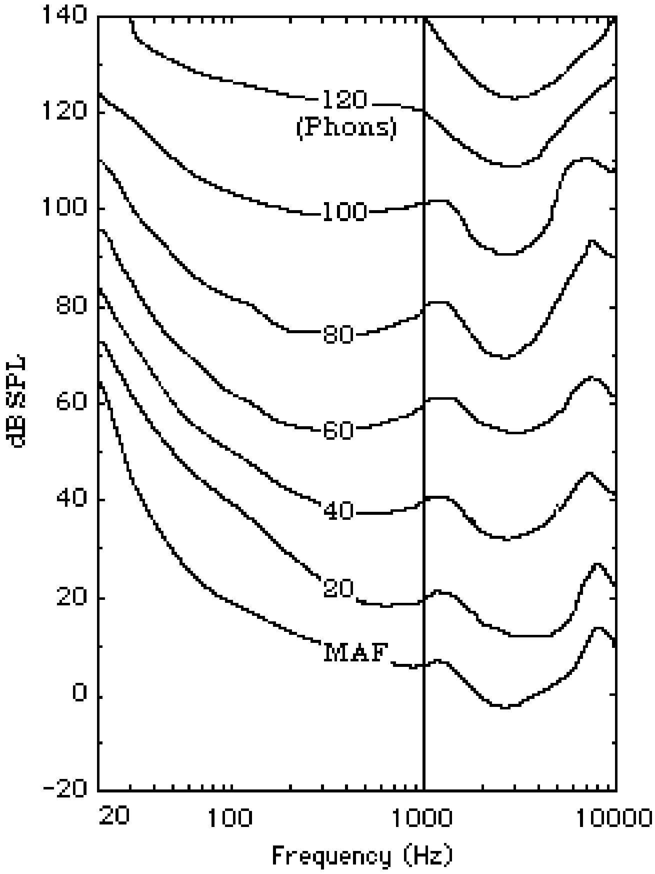


Figure 2.5: *Equal loudness contours (Phon curves) (from* [118]*).*

One other factor that is important in the design of auditory interfaces is that loudness if affect by duration [39]. For sounds of less than one second loudness increases with duration. This is important in auditory interfaces because short sounds are often needed so that the auditory feedback can keep pace with the interactions and they must be made loud enough for listeners to hear. Buxton *et al.* ([39], pp 2.10-2.11) also report that subjects are "very bad at making absolute judgements about loudness" and also "Our ability to make relative judgements of loudness are limited to a scale of about three different levels".

Masking

Another important factor is masking. The masking of one sound by another raises the threshold level of the masked one, therefore a greater intensity is needed to make the masked sound as loud as would be normally expected. The greatest masking effect will occur where the masked sound and the masker have similar frequencies. Sounds within a *critical band* mask each other more than those further apart. Critical bands are frequency regions within which sound energy interacts [39]. Within a critical band sound energy is summed, outside the band loudness is summed (for more detail on this see Moore [118]).

So, if information is to be presented the sounds used must be loud enough to be heard over any background masking noises otherwise the listener will miss information. Patterson [128] suggests that sounds should be between 10dB and 15dB above the threshold imposed by masking noises in the environment for them to avoid masking but without being too loud. He also indicates that sounds should contain many component frequencies as there is then less chance of the masker's frequencies being similar to the masked one's. This again shows the potential advantage of complex tones over simple ones; with only one component frequency there is more chance of two sounds masking each other. Pulsing a sound can also reduce the chance of it being masked. For much greater detail on this subject see Gelfand ([80] Chapter 10).

2.4 TIMBRE

As mentioned above, timbre is the 'quality' of a sound. This is a very difficult component of sound to define. It is often described as everything that is not pitch nor intensity [27]. As Blattner, Sumikawa and Greenberg [25] (p 26) say: "Even though timbre is difficult to describe and notate precisely, it is one of the most immediate and easily recognisable characteristics of sound".

Timbre is multidimensional but not all of the dimensions are known. Much work has been done on the identification of the attributes of timbre but no complete description has yet been produced. It may be the case that there are different descriptors for different types of sounds, for example one set for musical instruments, another for environmental sounds and another for speech. Two main groups of characteristics, however, have been identified: Spectral components and temporal components. These are the main areas studied by almost all researchers into timbre; Some have concentrated more on the temporal (Grey [86] and Plomp [135]) others on the spectral

(Wedin & Goude [176]). Von Bismarck [170] looked at the spectral components of sounds. He identified two important characteristics of timbre:

- ❖ Sharpness: Defined by the energy concentration of the frequency spectrum. Sounds with much energy at the high end of the spectrum sound sharp. This attribute is often called brightness.

- ❖ Compactness: Determined by the distinction between discrete harmonics of tones and the continuous harmonics of noise. This can be used to differentiate between tones and noise.

2.4.1 Models of timbre space

Grey [86] created a model of a timbre space for musical instruments based around a mainly temporal model. This is shown in Figure 2.6. The space has three dimensions or attributes:

- ❖ Energy distribution: Similar to the spectral component sharpness mentioned above.

- ❖ Static versus dynamic tone quality: This attribute characterises sounds with upper harmonics that enter, reach their maxima and decay at the same times and sounds with upper harmonics that have differing patterns of attack and decay.

- ❖ The final attribute is more difficult to define. One end represents instruments with low amplitude and high frequency energy in the initial stage of the attack, for example a clarinet. These instruments have a buzzy, soft attack. The other end of this dimension represents instruments that have dominant lower harmonics and more explosive initial attacks, for example a trumpet.

Wessell [181] created a simple computer simulation of a timbre space using two dimensions; One dimension was the shape of the spectral energy distribution, the other was attack rate or amount of synchronicity between the components. He could move around the space and create a new timbre for the current position. None of these examples is close to capturing the complex, multidimensional nature of timbre. A more complex, but still incomplete, model might contain some of the following dimensions:

Spectral Dimensions:

- ❖ Sharpness
- ❖ Compactness
- ❖ Type of harmonics (odd/even)
- ❖ Intensity of upper harmonics (decreasing or increasing)

Temporal Dimensions:

- ❖ Synchronisation of harmonics (dynamic tone quality)
- ❖ Speed of attack

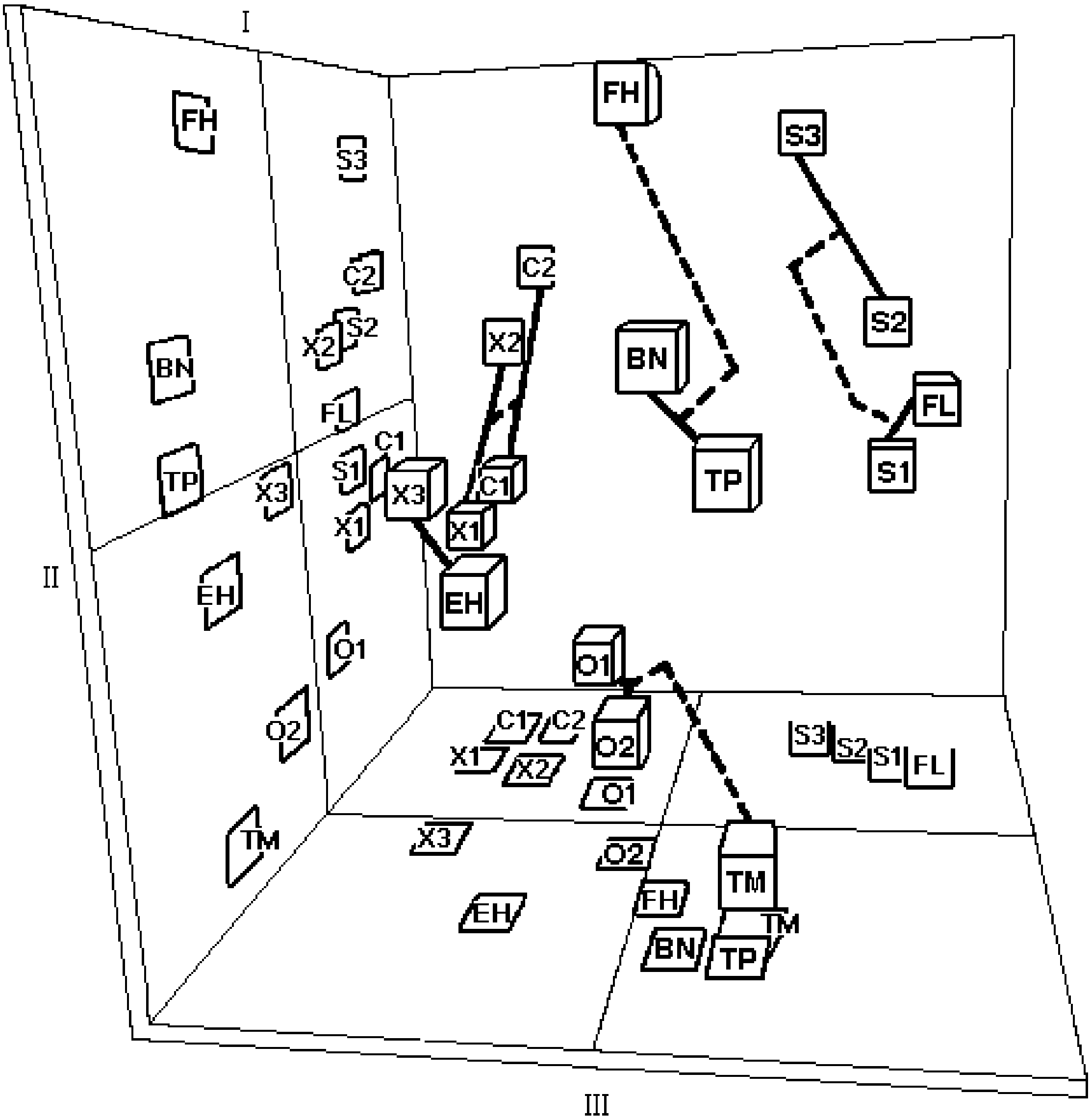


***Figure 2.6:** Timbre space as defined by Grey* [86].

A three dimensional representation of the similarities among 12 instruments. Instruments that are highly related are connected by solid lines, and instruments that are less strongly related are connected by dashed lines. The values on each of the three dimensions can be determined by looking at the position on the 'floor' and the position on the 'left' wall. Key: O1, O2 = oboes; C1, C2 = clarinets; X1, X2, X3 = saxophones; EH = English horn; FH = French horn; FL = flute; TM = muted trombone; TP = trumpet; BN = bassoon; S1, S2, S3 = cellos (from Grey [86]).

Other experimenters (Berger [17] and Grey & Gordon [87]) have tried to modify the temporal and spectral shape of musical instrument timbres. Subjects then made incorrect judgements of the type of timbre. Berger (p 1891) noted that: "Where tones were incorrectly identified under any conditions, it was noted that this incorrect identification was likely to be to a related instrument". This has important benefits for the use of timbres as it may mean there is some tolerance built-in and if, for some reason, the listener could not identify the timbre completely he/she would still be in the right group of instruments.

There is, as yet, no way to fully specify timbre. This causes problems as there is no formal way to describe how close one timbre is to another, it must be done subjectively. There is no way of saying if two timbres can be differentiated. This can lead to problems when trying to decide what timbres to use in an auditory interface; the only way to decide is to try some and see. To control the components of timbre described in this section requires the manipulation of very complex and low-level sound parameters which is very difficult. Researchers are beginning to find some structure within timbre (see Lerdahl [107]) but little has been done in the area yet. Lerdahl tried to organise hierarchies of timbre but did not develop any descriptors for it. One method to overcome this may be to use vocal segregates (see Avons, Leiser & Carr [10] and Leiser, Avons & Carr [106]). These are short non-lexical utterances (such as ‘mm-hmmm’) which occur in natural dialogue. Avons *et al*. claim that they are easily recognisable. The problem with them may be that there is only a limited set of segregates and they may be hard to manipulate whilst remaining recognisable. This is an area that requires further investigation.

The lack of a set of descriptors of timbre means that timbres cannot be varied in a systematic way by changing the descriptors. If there was such a set sounds with one timbre could be transformed into sounds with another timbre and two sounds with the same timbre could be made to sound just noticeably different from one another. These difficulties make timbre difficult to use precisely in an auditory interface.

In order to generate different timbres an electronic musical instrument synthesiser may be used. Even though some are very sophisticated, the quality of the synthesis may mean that many of the timbres available are difficult to differentiate and difficult to recognise as particular instruments. In many cases, instruments that sound very different in the real world become indistinguishable when played on a synthesiser because of the quality of the synthesis. This causes problems if, in an interface, users are expected to be able to remember a timbre and link it to an interface object. For example, if they had to remember that a piano sound meant wordprocessor but they could not differentiate the piano from the other timbres then the system would be unusable. Synthesisers usually provide good control of pitch, loudness, stereo position, etc. but almost no ability to change the characteristics of a timbre (without changing to a different one). Work has been done by Oppenheim, Anderson & Kirk [127] to address this problem. They discuss two systems that allow a composer to specify the perceptual parameters of a sound, rather than just pitch or duration. These systems are still research prototypes and are not yet available on commercial synthesisers.

2.5 LOCALISATION

Another important factor in the perception of sound is spatial information. Localisation is the ability to identify the position of a sound source in space. This ability is important for humans and animals as it determines the location of objects to seek or avoid, and the direction for visual attention. If a sound source is located to one side of the head (see Figure 2.7), then the sound reaching the further ear will be reduced in intensity (Interaural Intensity Difference - IID) and delayed in time (Interaural Time Difference - ITD). This is known as the *Duplex Theory* and was first developed by Lord Rayleigh (1907, from Moore [118]). See also Levitt & Voroba [108] for more details.

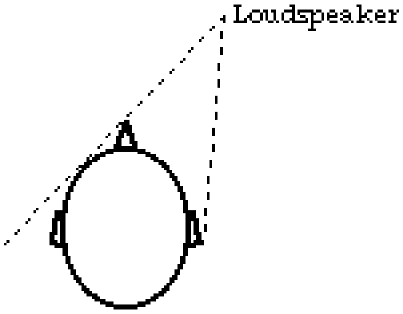


Figure 2.7*: Position of the loudspeaker in relation to the head (from* [80]*).*

2.5.1 Interaural Intensity Difference

At low frequencies sound waves bend around the head as their wavelength is large compared to the size of the head and there is little interaural intensity difference (Figure 2.8). At higher frequencies (Figure 2.9) sound waves do not bend (as the wavelengths are short) so that an auditory shadow is created and there is a greater IID. Moore ([118] p 195) suggests that there can be an intensity difference of up to as much as 20dB across the ears. It can be seen that IID has little effect at low frequencies but is much more important for localisation at higher frequencies - above 3000Hz.

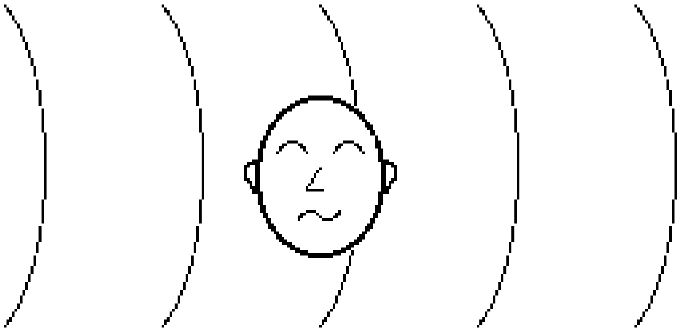


Figure 2.8*: Low frequency sounds bend around the head (from* [80]*).*

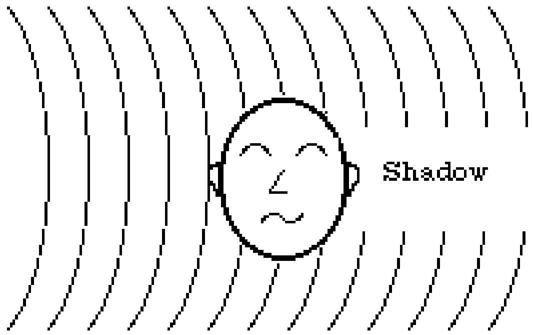


Figure 2.9: *High frequencies have wavelengths smaller than the diameter of the head and an auditory shadow occurs (from* [80]*).*

2.5.2 Interaural Time Difference

Figure 2.10 illustrates interaural time difference. When the sound source is at 90° to the head, i.e. opposite one ear, there is a greater than 0.6 msec delay between the ears. The sound reaching the nearer ear leads in phase and it is this that the ear uses to locate the sound. As frequencies increase to above 1500Hz, the wavelength is shorter than the distance between the ears so that phase information becomes unreliable. Therefore, ITD is only useful for localisation at lower frequencies. Delays of up to 2 msecs can be heard but longer than this and the sound tends to be heard as two separate tones. See Scharf & Houtsma [147] for more detail on both IID and ITD.

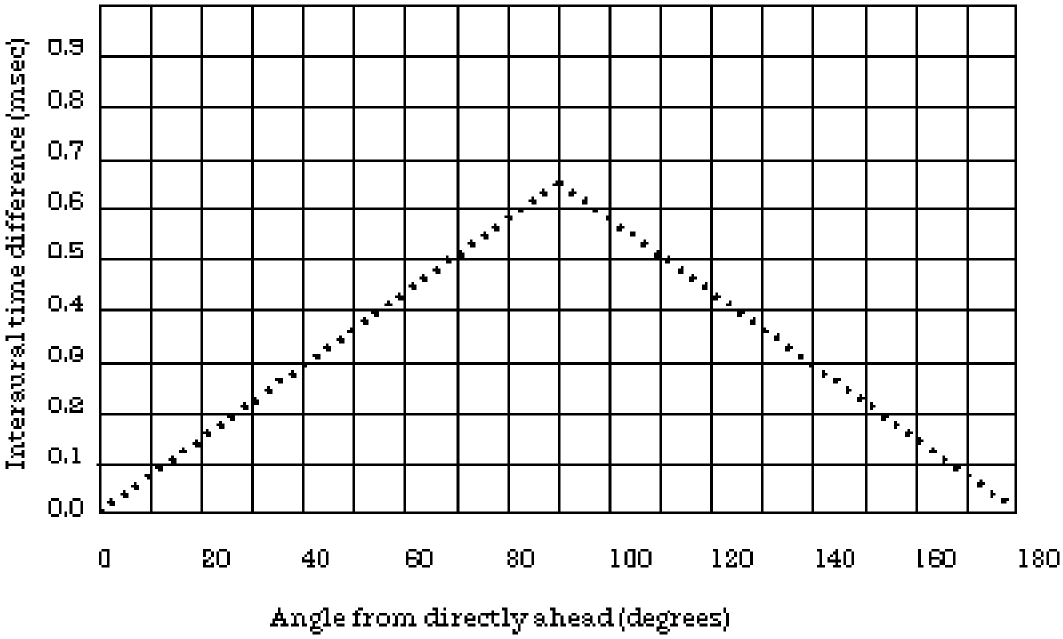


Figure 2.10: *Interaural time differences for different loudspeaker positions (from* [80]*).*

IID and ITD can be used in auditory interfaces to provide directional or positional information. Pitt & Edwards [134] have shown that using IID a user can find targets on an auditory display accurately and with reasonable speed.

Humans can detect small changes in the position of a sound source. The minimum auditory angle (MAA) is the smallest separation between two sources that can be reliably detected. Strybel, Manligas & Perrott [161] suggest that in the median plane (the median plane cuts through the head vertically, along the line of the nose) sound sources only 1° apart can be detected. At 90° azimuth (directly opposite one ear) sources must be 40° apart. This has important implications for auditory displays. It indicates that 'high-resolution' sounds can be used when presented in front of the user.

2.5.3 Stereo sound, lateralisation and localisation

Much of the recorded sound we hear is in *stereo*. As Burgess [37] describes, along with the sound, a stereo recording captures differences in intensity. From these differences the listener can gain a sense of movement and position of a sound source in the stereo field. The perceived position is along a line between the speakers. This simple, inexpensive technique can give useful spatial cues at the auditory interface (see [134]).

In many psychoacoustic experiments headphones are used to supply the stereo sound to the listener for easier control. A listener wearing headphones uses *lateralisation* to locate the position of a sound which will be perceived as being on a plane between the ears and ‘within’ the head. Burgess ([37], p 1) suggests that this is because:

> “…the microphones used for stereo recording provide a poor model of the way sound really arrives at the ears. Human ears are not several feet apart, they do not have symmetric field patterns and they are not separated by empty space”.

Work has been done by Sakamoto, Gotoh & Kimaura [146], Wenzel, Foster, Wightman & Kistler [180], Gerhing & Morgan [81] and Wenzel, Wightman & Foster [178] and Wenzel [180] to make sounds perceived through headphones appear fully in three dimensions, or ‘outside’ the head. Wenzel developed a system to allow the production of three-dimensional sounds for virtual reality systems so that sound could be presented as coming from anywhere in space. To do this, the sounds entering the ear are recorded by putting microphones into the ear canals of listeners. This is obviously a much better model of how sounds actually arrive at the eardrums than used in stereo recording. The differences between the sound at the sound-source and at the eardrum are calculated and the ‘head-related transfer functions’ derived are used to create filters with which stimuli can be synthesised. Bergault & Wenzel ([16], p 7) suggest that: “When stimuli recorded in this way are played back over headphones, there is an immediate and veridical perception of 3-D auditory space”. This research is important as three-dimensional auditory interfaces can be created with the user moving in all dimensions and hearing objects from anywhere within the interface relative to them. Cohen [44] has built systems using three-dimensional sound. These are described in the next chapter.

The cost of producing spatialised sounds using high-performance systems, such as the Convolvotron (produced by Crystal River Engineering [69]), is high (approximately £1000 at this time). The hardware must be extremely powerful to be able to perform the calculations necessary in real-time. One way around this is to simplify the filters used and run them on Digital Signal Processor (DSP) chips. These are much cheaper and are becoming standard on some machines [154, 175, 185]. The drawback is that the quality of the spatialisation is reduced and listeners might hear the sounds as within their heads or be unclear as to the direction of the source. Burgess [37] has attempted to do this type of spatialisation. He has shown that sampling rates can be reduced from 50kHz (in the Convolvotron) to 22.05kHz and filter lengths reduced from 512 samples to 32 samples and for perceived spatialisation to be retained. Lower sampling rates and filter lengths caused a loss of perceived spatial location. These reductions allow spatialisation to be performed on much cheaper hardware and make it available for use in interfaces in general.

2.6 AUDITORY PATTERN RECOGNITION

Sounds are not normally perceived purely in terms of pitch, intensity and timbre but rather in terms of discrete sound sources, or *streams*, each of which has its own pitches, intensities and timbres. When sounds are perceived as sources higher level structures can be used to identify them. Bregman [30], Williams [183] and Deutsch [52] put forward some factors by which sounds are grouped into separate sources. There are two broad groups of these.

2.6.1 Perceptual factors for pattern recognition

Firstly, some perceptual factors will be considered for the grouping of sounds into sources:

- ❖ *Similarity and dissimilarity:* Components which share the same attributes will be perceived as related and vice versa. Sounds will be grouped as a single source if they are similar, i.e. they have similar pitches, timbres, loudnesses and locations.

- ❖ *Proximity:* Components which are close in time and frequency will be perceived as related and vice versa.

- ❖ *Good Continuation:* One property of sound sources is that changes in pitch and intensity tend to be smooth and continuous. Sudden changes imply new sources have become active.

- *Coherence:* Components of streams change in coherent ways. Linked with good continuation is the fact that components of a source also tend to vary in a coherent manner, for example they change intensity and pitch together. An example of this is modulation, where frequency or intensity are varied in a regular manner to cause a sound to split into different sources (see Figure 2.11). In Figure 2.11 (A) several harmonics are being played and are heard as one source.

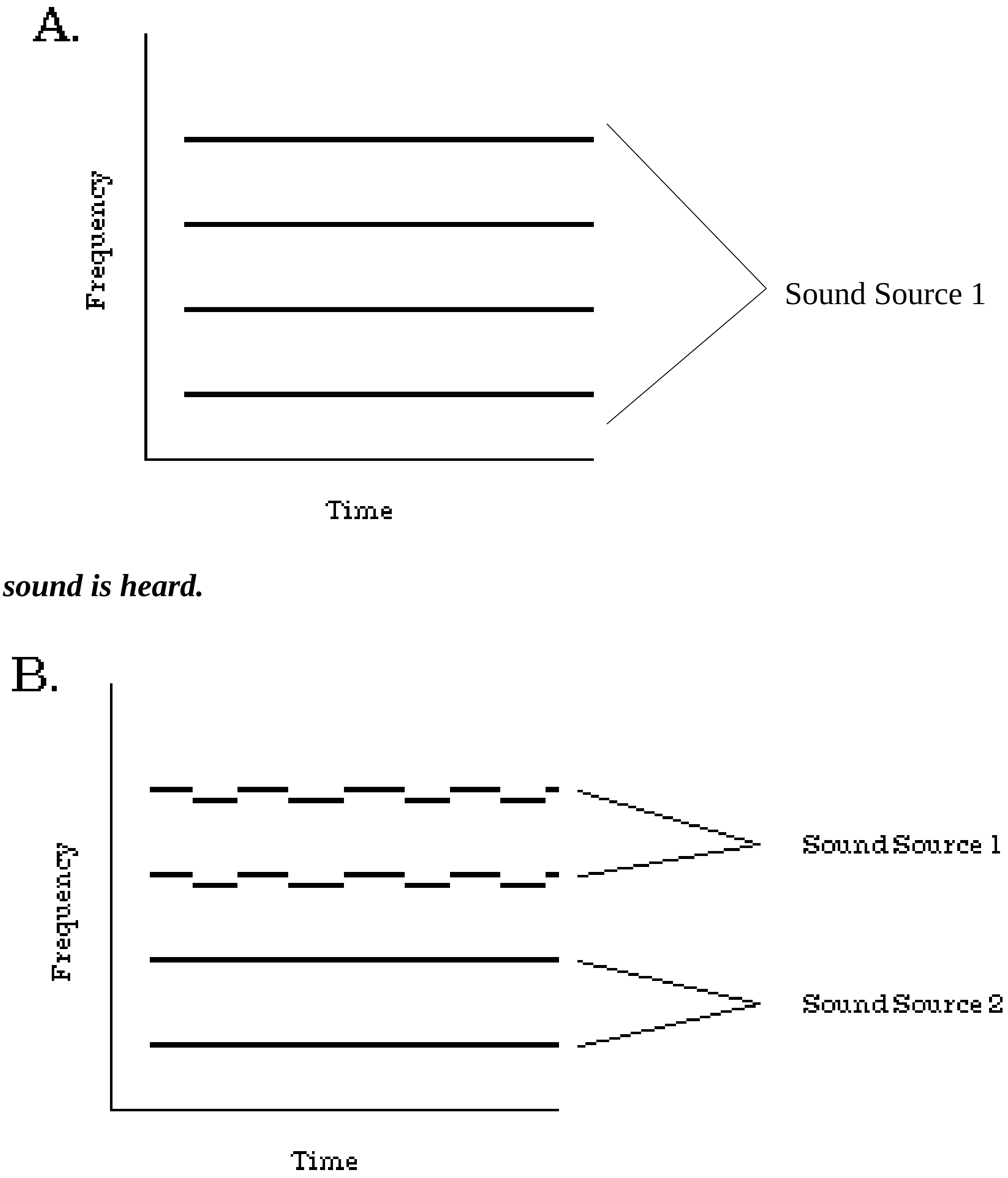
  In Figure 2.11 (B) two harmonics are frequency-modulated and this causes them to be heard as a separate source. In music, modulation takes the form of tremolo or vibrato and can be used when playing an instrument to help a listener identify a single instrument within a group of players by causing each instrument to be heard as a separate source.

2.6.2 Physical factors for pattern recognition

Certain physical factors will also cause grouping of sounds into sound sources:

- *Fundamental Frequency:* If two sounds with different fundamental frequencies are heard the harmonics are not confused (see section 2.2.3 on the missing fundamental): they will be perceived as separate sources. If the fundamentals are the same then fusion occurs and the sounds become one source.

- *Sound Location:* Sounds originating from similar locations in space will be perceived as sources. Buxton, Gaver & Bly [39] suggest that spatially separating sound sources helps clarity by allowing the auditory system to focus on a single source from amongst many and lessens interference between sounds. The ‘Cocktail Party Effect’ [9] shows that it is easier to listen to two sounds if they occupy different positions in space, rather than a common one. At a cocktail party a listener can follow many different conversations within the room without confusion if they occur in different locations. This effect could be used in auditory interfaces to allow the user to monitor simultaneous sounds sources without confusion. This is supported by research from Mayfield (reported in Gerth [82]) where, without spatial separation of sources, recognition rates fell more rapidly as sound density (number of sources) increased than when separation was used.



One sound is heard.

Two sounds are heard.

***Figure 2.11:** The effects of coherence*

❖ *Rhythm:* Rhythmic patterns tend to be perceived as sources [49]. According to Deutsch rhythm is one of the most powerful physical factors of pattern recognition. This importance was noted by Blattner, Sumikawa & Greenberg [25] in their work on earcons (see next chapter, section on earcons) where rhythm is one of the main differentiating factors they use. Moore suggests that, for complex tones, a gap between tones of only 2-3 msecs is detectable. Handel ([88], pp 388-389) suggests some methods used for creating rhythmic groups (see also [71]). For example:

- *Intensity accentuation* - an accented note begins a group. If the intensity of every second or third note is increased then a sequence of notes will be perceived as groups of two or three notes.
- *Duration accentuation* - a long note ends a group. If every second or third note is lengthened then a sequence of notes will be perceived as grouped into two's or three's with the long notes at the end of each group.
- *Interval differences* - a long interval every two or three notes causes grouping into two's or three's.
- *Frequency differences* - pitch affects perceived grouping. High pitched items tend to be perceived as accented item in a group.
- *Natural groupings* - it is natural to group notes in two's, three's or four's, five or seven note groups are harder to perceive.

Handel suggests that these parameters can be played off against each other to create the complex rhythms used in music.

❖ *Scale and Key Structure:* Listeners use their knowledge of scale and key structures to group patterns of tones. Dewar, Cuddy and Mewhort [53] found that subjects could judge the differences between two patterns of tones very accurately if all the notes from the first were in a different scale to the notes of the second. People can detect a note in a different key to a group of notes very easily.

These factors are important when creating auditory interfaces. The designer of an interface must make sure that the user interprets the sounds presented correctly; that the sounds confer to the listener the information that the designer wants them to. If a designer wants to present some information as coming from a single source - for example, to represent an object within an interface, he/she must make sure that the listener perceives it as a source by using the factors mentioned above. If it is not perceived correctly the information will be lost or misunderstood. The grouping of sounds into sources may also allow the user to monitor more simultaneous sounds than would otherwise be possible, by virtue of the cocktail party effect.

These factors are also extremely important if multiple sounds are to be played together but are to be heard as coming from different sources. For example, if each process in a multiprocessing environment has a different sound the user must be able to hear them as separate sources otherwise a cacophony will result and no information will be gained from the sounds. Later in the thesis these factors will be used to help differentiate two different complex sounds playing at the same time.

2.7 CONCLUSIONS

This chapter has given an introduction to the study of the perception of sound, or psychoacoustics. The mechanisms behind the perception of each of the components of sound have been described along with factors that affect the perception. The designer of an interface must be aware of the range of human auditory processing so that any interface that is created can be used by a listener. This review of psychoacoustics will be used in Chapters 4 and 5 to create the earcons used in the experiments. A set of guidelines is proposed containing information on psychoacoustics and the results from the experiments. Interface designers who then want to use sound in their multimodal interfaces will not have to have detailed knowledge of sound perception in order to create effective interfaces, they can just use the guidelines.

CHAPTER 3: NON-SPEECH AUDIO AT THE HUMAN-COMPUTER INTERFACE

3.1 INTRODUCTION

This chapter provides some examples of existing systems that use non-speech audio in their human-computer interfaces. It begins with a discussion of why non-speech sounds rather than synthetic speech should be used when sonifying an interface. There are some serious drawbacks with speech that make non-speech audio more attractive. Psychological research is described that shows the advantages which can be gained from using multimodal interfaces. The next section investigates problems that can arise if the volume range used for audio feedback in the interface is not controlled. Some of the problems of using sound in the interface are then reported. It is argued that many of these problems would not occur if carefully designed non-speech sounds were used. Then brief examples of some existing auditory interfaces are given. Finally, several of the most important examples of auditory interfaces are described in greater detail. The interfaces considered here use a mixture of graphical and non-speech audio for output and the standard keyboard and mouse for input.

This thesis is concerned with interfaces that use sound to enhance human-computer interaction, rather than using sound to display data. That is the field of scientific auralisation, an extension of visualisation. Groundbreaking work was done in this field by Bly [27] who added sounds to multi-dimensional military battle data. As in the case of auditory interfaces, this is a new and growing area. For a review see Iverson [96] or the proceedings of ICAD'92 [102]. These two areas are closely linked and results from one may be applicable to the other. However, the review presented here concentrates on the use of sound to enhance and enrich the human-computer interface.

3.2 WHY NOT USE SYNTHETIC SPEECH FOR OUTPUT?

The interfaces described in this thesis use non-speech sounds. Why not use synthetic speech? Presenting information in speech is slow because of its serial nature; to assimilate information the user must hear it from beginning to end and many words may have to be comprehended before a message can be understood. With non-speech sounds the messages are shorter and therefore more rapidly heard. Speech suffers from many of the same problems as text in text-based computer systems, as this is also a serial medium. Barker & Manji [15] claim that an important limitation of text is its lack of expressive capability: It may take many words to describe something fairly simple. Graphical displays were introduced that speeded up interactions as users could see a picture of the application they wanted instead of having to read its name from a list [15]. In the same way, an encoded sound message may be able to communicate its information in fewer sounds. The user hears the sound then recalls its meaning rather than having the meaning described to them in words. The pictorial icon is also universal, it means the same thing in different languages and the non-speech sound would have similar universality.

Work has been done on increasing the presentation rate of synthetic speech (see Aldrich & Parkin [1] and Slowiaczek & Nusbaum [153]). Both of these found that the accuracy of recognition decreased as the speech rate went up. Slowiaczek & Nusbaum found that (p 711) “a speaking rate of about 150 words/min. is optimal for perception of synthetic speech”. This figure is around the normal speaking rate and is very slow to listen to. When they increased the rate to 250 words/min. (the normal sight-reading rate) recognition decreased significantly. One of the main causes of this, they suggest, is the poor quality of the synthetic speech. Much of the prosodic information (intonation, pausing etc.) in normal speech is not given in synthetic speech. At low speeds listeners can cope without it but at higher speeds it becomes much more important. With these problems, users of synthetic speech (p 711) “will be constrained to operate at rates that are far below normal reading rates”. Highly-skilled users, such as visually-impaired people, can reach higher recognition rates but this requires much practice. This research shows that there are problems with using speech and these are not shared by non-speech sounds.

Smither [156] conducted an experiment to investigate the demands synthetic speech puts on short term memory. He tested natural speech against synthetic speech on young and old adults. His results showed that synthetic speech put a heavier load on short term memory than natural speech. Older subjects performed worse than younger ones but both groups performed worse with the synthetic speech. This experiment again points to problems with using synthetic speech.

One important ability of the auditory system is that of auditory habituation [39] where continuous sounds with a restricted loudness range can fade into the ‘background’ of consciousness after a short period of time. If the sound was to change (or stop) then it would come to the foreground of attention because of the sensitivity of the auditory system to changes in stimulus [38]. Habituation difficult to achieve with speech because of the large dynamic range it uses. According to Patterson ([128], p 11):

> “The vowels of speech are often 30 dB more intense than the consonants, and so, if a voice warning were attenuated to produce a background version with the correct vowel level the consonants would be near or below masked threshold”.

As will be discussed later, continuous sounds can be used to provide monitoring information on processes within a system. Speech could not be used for this as it cannot fade into the background of consciousness and also each process to be monitored could not continuously speak its name and status without problems of masking. The quality of most speech synthesisers is also poor (although they are getting better all the time) which may cause problems with understanding unless users are well trained.

The use of speech may also be annoying for other users who overhear the interface. Hapeshi & Jones ([89], p 91) say:

> “A number of studies have demonstrated that these attention grabbing qualities of the auditory channel, combined with its special adaptation to speech, makes it highly disruptive when listeners are performing some verbal activity”.

They also give evidence to show that background speech, even at low intensities, is much more disruptive than non-speech sound when recalling information. Baddeley ([11] p 74) reports the ‘unattended speech effect’. Unattended speech, i.e. in the background, causes information to be knocked out of short-term memory, whereas noise or non-speech sound does not even when it is pulsed to give the same intensity envelope as speech. This problem is unaffected by the intensity of the speech, provided that it is audible. This shows a problem for speech at the interface as it is likely to prove disruptive for other users in the same environment unless it is kept at a very low intensity. We saw above that this can cause problems with the ability to hear consonants.

Baddeley also reports an experiment to test the effect of music, both vocal and instrumental, to see if it had the same problems as speech. The results of the work showed that vocal music exhibited exactly the same problems as unattended speech. However, with instrumental music the effect was greatly diminished. These results seem to indicate that non-speech audio has great potential at the interface and is less likely to be a problem for others listening nearby.

These examples show there are many compelling reasons to use non-speech audio rather than speech. The fact that speech may be disruptive for other users who can hear the interface and may also disrupt the work of the principle user of the computer if they are trying to perform a recall task are strong reasons to use non-speech sounds. The following section will discuss the advantages to be gained from multimodal interfaces that use both graphics and sound.

3.3 THE PSYCHOLOGICAL BASIS FOR MULTIMODAL INTERFACES

Psychological evidence shows that there are advantages to be gained from using combined auditory and graphical multimodal interfaces and that work in their development is justified. Some early work was done by Colquhoun [46] in which simple sounds were added to a visual sonar monitoring system. Users had to monitor either an auditory, a visual or a dual-mode display and indicate the presence of a target stimulus. The test results showed that the audio-visual display got the highest target detection rates, although error rates were similar. Colquhoun suggested (p 425):

> “Dual-mode displays with ‘redundant’ signals (i.e. signals simultaneously presented on auditory and visual displays) have been found superior either to both single-mode displays … or to at least one of them. In no case has the dual-mode display been found significantly *inferior* to a single-mode display”.

This research indicates that there are definite advantages to be gained using dual-mode displays.

Brown, Newsome & Glinert [36] performed visual search experiments using auditory or visual target cues. Their aim was to reduce visual workload by using multiple sensory modalities as suggested by the Multiple Resource Theory (for more information on the theory see [182]). They define the theory thus (p 340):

> “…humans possess different capacities, each with separate resource properties. If tasks demand separate resources, performance of two simultaneous tasks will be more efficient … task interference occurs when the same resources are called upon simultaneously”.

They suggest that *intermodal* task sharing (i.e. dividing attention to a task between eyes and ears) is more successful than *intramodal* (i.e. the eyes doing two things at once). The experiments they conducted showed that the auditory modality could, in some cases, be more effective than the visual one. Their results also indicated that the auditory modality was not as fast although they suggest that (p 345) “Extracting information from an auditory cue is foreign to most people. It is possible that with a longer training session this difference would no longer exist.”. However, they did not conduct any experiments to see if this actually occurred. Nonetheless, their findings do suggest that humans can extract more than one piece of information from a sound and then act upon it.

Work by Perrott, Sadralobadi, Saberi & Strybel [132] showed that providing auditory cues can help in the location of visual targets on a display. They used three-dimensional sound (see the section in the previous chapter on localisation) to indicate the position of a target. The target was sometimes the only stimulus on the display, at other times many distractors were present to make the target more difficult to locate. The target could lie within the central visual field or be outside it. Their results were favourable (p 398):

> “The presence of spatial information from the auditory channel can reduce the time required to locate and identify a visual target even when the target occurs within a restricted region around the initial line of gaze”.

Even when the target was close to a subjects’ focus of visual attention he/she still located it more rapidly with a sound cue present. Perrott *et al.* go on to say:

> “The advantage of providing auditory spatial information is particularly evident when a substantial shift in gaze is required in the presence of a cluttered visual field”.

It is often the case with complex graphical interfaces and large, multiple-monitor displays that such situations occur and this research indicates that sound would be very effective in increasing performance with such systems.

The visual system is often considered to be the most important sense, dominating the others. If this were invariably true then auditory interfaces could not provide information more efficiently than visual ones and any extra data to be displayed would best be done through the visual sense. It is not, however, always the case that the visual system dominates the auditory. Walker & Scott [172] found that humans judge one light as being of a shorter duration than another identical one when a tone is played. They conclude (p 1327) “Thus auditory dominance occurred under the preceding conditions – that is auditory-visual conflicts in perceived duration…were resolved in favour of the auditory modality”. They suggest that the auditory modality may be a more appropriate means of processing information in the temporal domain and the visual modality in the spatial domain. Pezdeck [133] carried out experiments to determine if the visual sense dominated the auditory one in the comprehension of information presented on television. Whilst there was evidence of visual dominance, the presence of the auditory channel improved comprehension. This again shows that there are definite benefits to be gained from presenting data in an auditory form.

O'Leary & Rhodes [125] showed that if there is ambiguity in one mode then information presented in another can help to resolve it. So, for example, if a large amount of visual information needed to be presented sound could be used to help the user assimilate it. Loveless, Brebner & Hamilton [109] showed that the threshold level of vision could be raised by simultaneously presenting the same information to the auditory system (and vice versa). This, for example, could aid the comprehension of low-quality visual information. Watkins & Feehrer [174] have also done work in this area.

The above factors show that there are many potential advantages to be gained from adding sound to human-computer interfaces. This gives a strong base from which to work. A note of warning is sounded by Portigal [136]. He conducted an experiment to investigate the extraction of document structure using graphics, sound and a combination of both. The results showed that in the graphical and combination conditions the results were the same but more effort was required in the multimodal condition. The work of Wagenaar, Varey & Hudson [171] showed that combining different modalities does not necessarily have a beneficial effect. They tested different combinations of pictures of objects and words (names of objects) to discover what effect presenting information in different modalities would have. They concluded that multimodal presentation led to poorer recall. They say the gain that can be obtained by presenting the same information in two modalities is smaller than can be expected from presenting the same information in the same modality twice. This work was based around speech stimuli rather than non-speech but the problems may still apply. It indicates that further investigation of multimodal interactions is necessary in order to find the best way of combining sound and graphics. Sound cannot simply be added in an *ad hoc* way and improvements expected, research must be undertaken to find out how best to combine modes.

3.4 SOME PROBLEMS WITH NON-SPEECH SOUND AT THE INTERFACE

So far the advantages that can be gained from adding sound to human-computer interfaces have been discussed. Sound does have difficulties and limitations. Kramer [103] suggests some of these are:

- ❖ *Low resolution:* Many auditory parameters are not suitable for high-resolution display of quantitative information. Using loudness, for example, only a very few different values can be unambiguously presented. Vision has a much higher resolution. The same also applies to spatial precision in sound. Differences of about one degree can be detected in front of the listener and this falls to ten or fifteen degrees to the side [179]. In vision differences of an angle of two seconds can be detected in the area of greatest acuity in the central visual field.

- ❖ *Presenting absolute data is difficult:* Many interfaces that use sound to present data do it in a relative way. A user hears the difference between two sounds so that they can tell if a value is going up or down. It is very difficult to present absolute data unless the listener has perfect pitch (see Chapter 2). In vision a user only has to look at a number or graph to get an absolute value.

- ❖ *Lack of orthogonality:* Changing one attribute of a sound may affect the others. As discussed in Chapter 2, changing the pitch of a note affects its intensity and *vice versa*.

These problems must be born in mind when creating auditory interfaces. There is nothing that can be done about them, interfaces must be designed around them. Some of the other drawbacks of sound will now be discussed. In his paper, Jones [99] puts forward several reasons why sound should not be used at the interface. This thesis argues that these reasons are not valid in the case of non-speech sounds.

3.4.1 Sound is attention grabbing

As Jones says (p 383):

> “When we first consider the functional organisation of auditory perception in the brain, one of the most noticeable features is that audition is intimately connected with the arousal and activation systems of the nervous system”.

He calls it the ‘sentinel of the senses’ as it is well adapted to detect transients. He suggests that it should be used as a modality for warning signals and then used only rarely. This thesis suggests that sound can be used for messages other than warnings. Continuous sounds can present information and can fade into the background of consciousness. If sounds are carefully designed then they can be made less attention grabbing. Edworthy, Loxley, Geelhoed & Dennis and Edworthy, Loxley & Dennis [64, 65] have shown that the perceived urgency of sounds can be predictably controlled (see the section on existing guidelines for the use of sound below for more details).

Jones also says that speech output may interfere with other tasks being undertaken (see the description of the ‘unattended speech’ effect given previously). Baddeley [11] shows that unattended speech can cause problems but that non-speech sounds do not suffer from the same problems. Jones also talks about the *Suffix Effect* [11] and the fact that speech can knock items out of short-term memory. Baddeley has shown that this is not a problem with non-speech sounds.

3.4.2 Sound is a public event

There are problems of noise pollution and privacy associated with sound output. For example, it is easy to tell when someone in the next room is having a problem with their computer because of the number of beeps heard. In general, the error beeps produced by computers are too loud. In this thesis it is argued that the sounds should be kept to just above the threshold of the ambient sound level. This is the case with the sound that a hard disk makes, the primary user can hear it but anyone else nearby is unlikely to. The other alternative is for users to wear headphones but this is not a satisfactory solution because users need to be able to talk to each other and hear things going on in the rest of the environment. The problems of noise pollution and privacy can be overcome by reducing the intensity level of the sounds used at the interface. An investigation into the annoyance caused by sound was conducted as part of the experiments described in Chapter 7.

3.4.3 Sound is transient

Many sounds in everyday life are not transient, as Jones suggests. Think, for example, of an air conditioning system: Its sounds continue for long periods of time and become habitual. Sounds often continue in the background and only become apparent when they change in some way. This can be taken advantage of at the interface and this thesis suggests uses for this type of sound. The next problem Jones describes is that sound presentation is serial and this places an extra burden on memory, as compared to visual presentation. Sounds in everyday life occur in parallel and there is no reason why this could not be exploited at the interface. Gaver, Smith & O’Shea [79], Blattner, Papp & Glinert [26] and Chapter 5 of this thesis show that multiple sounds can be used simultaneously to present information. In the previous chapter, auditory pattern recognition was reviewed and this provides mechanisms to allow different sound sources to be used in parallel.

3.4.4 The relation of sound to objects

Jones suggests that there is a problem mapping the sounds in an interface to the objects they represent. Chapters 4 and 5 of this thesis (see also Brewster, Wright & Edwards [32, 33, 34]) show that users can effectively recognise complex auditory cues and link them to their actions/operations in the interface.

Jones also talks about problems of anthropomorphism due to speech at the interface (p 386). He says there is a “tendency for people using speech systems to regard the device as having greater power and flexibility than comparable to visual/manual systems”. This is not a problem for non-speech audio output as it has none of the characteristics of human speech. The problems put forward by Jones have been answered. Most of these are difficulties faced by using synthetic speech and the careful design of non-speech audio feedback can avoid most of the drawbacks he suggests.

3.5 THE PROBLEM OF LOUDNESS AT THE INTERFACE

Loudness is one of the most easily controlled parameters of sound. Overall control is normally given to the user by a volume knob so that he/she can adjust it as necessary. Care should be taken when using it at the interface. The loudness of a sound can be thought of as similar to the brightness [141] on a video monitor. On a monitor the user can change the brightness of the display in response to the ambient light level. If the room is light then the brightness of the display will be increased so that the information on the screen can still be seen. If the room is dark then the brightness will be turned down so that the screen does not hurt the eyes. The volume control on a monitor acts in a similar way. If the room is noisy then the loudness will be increased to avoid masking. If the room is quiet then the loudness will be reduced to avoid irritation. If the sounds used at the interface vary widely in loudness then turning up the volume so that the quiet sounds can be heard will cause the loud sounds to become irritating. Conversely, turning down the loud sounds to a pleasant level may cause the quiet ones to fall below the threshold of hearing. This is a similar problem to that described in the previous section about the perception of vowels and consonants. It indicates that the volume of sounds in the interface should be kept within a narrow range so that overall adjustment does not cause any to be lost. Annoyance is also closely linked to loudness. Loud sounds are very likely to be annoying to the user and others nearby (the topic of annoyance is dealt with in detail in Chapter 7).

3.6 A REVIEW OF SOME EXISTING AUDITORY INTERFACES

A brief description will now be provided of some auditory interfaces before four examples are discussed in greater detail. Some of the earliest work in this area was the addition of sound to the output of a seismometer [158]. The output was time-compressed and experiments were carried out to see if subjects could tell the difference between earthquakes and underground nuclear bomb tests. Commenting on the results, Speeth ([158], p 913) says “Listeners were successful in separating one class of events from the other in over 90% of the cases presented to them”. These results show that listeners can make accurate judgements on very complex sound patterns.

Other early work was the addition of sound to the interface of a CAD system [2]. The task for evaluating the usefulness of the auditory feedback was to design the acoustical treatment for a noisy computer room. Partitions had to be positioned in the room to absorb the sound. One group of subjects got no audio feedback, just visually presented data, the other group got sound feedback on the amount of noise reduction a partition would give along with the visual data. As the experiment chosen was based around sound providing auditory feedback was relatively straightforward. If a different experiment had been used (for example, the design of a nut or bolt) it might have been more difficult to provide useful sound feedback. Nevertheless, the results showed that the audio feedback helped to reduce the number of errors and to produce a quieter computer room.

The auditory navigation work carried out by Pitt & Edwards [134] used the concept of localisation (see the previous chapter) to provide the spatial information necessary to allow a visually disabled user to navigate his/her way around an auditory display and select auditory objects (simulating the way a sighted user moves around a graphical screen and selects icons). Each target on the screen had a different sound, for example a different timbre and pitch, or speech and they were all played concurrently. Various cues were used to provide directional information to the user. Intensity differences between the ears were used in the form of stereo information. These gave the users cues as to where the target was in relation to them in the horizontal plane. Intensity cues gave information about proximity to a target: The closer a user was to a target the louder that target’s sound was. Pulsing of sound sources helped the subject differentiate different targets (see the previous chapter on auditory pattern recognition). Results from this work have shown that users can find up to eight targets on the screen with reasonable speed and accuracy. Expensive spatialisation hardware (see previous chapter on localisation) is therefore not always necessary in order to provide strong spatial cues to listeners.

DiGiano & Baecker [55] and DiGiano, Baecker & Owen [56] integrated non-speech audio into a Logo programming environment. DiGiano *et al.* [56] say (p 301): “The LogoMedia programming environment supports the ability to associate non-speech audio with program events while the code is being developed”. They provide a system of audio probes which can be attached to sections of code and then triggered to perform some action. *Control Probes* can be used to monitor control or data flow. A probe is added to a statement and a sound is then triggered when the statement executes. These can also be added to procedure entry/exit so that programmers can detect if code is being executed in the sequence expected. The actions that can be triggered can play sampled sounds, turn on musical instrument synthesisers or change the volume or pitch of a tone. Different instruments, for example, can be used on different triggers. *Data Probes* can be used to monitor variables. Pitch could be attached to a variable in a loop so, as the variable increased in value the pitch of the tone would rise. If the pitch did not rise or kept rising too far then the error would be easily identified. DiGiano *et al.’s* papers give other examples of using probes to discover errors. This system allows the programmer a very powerful method of debugging code.

In his thesis, DiGiano [54] showed results of experiments to test the effectiveness of the audio probes in LogoMedia. The probes were used by the subjects in preference to simple writes to the screen. He says (p 55):

> "All three subjects took advantage of sound to increase the bandwidth of communications between program and programmer. Non-speech audio was used by subjects to keep them aware of program events and values while their visual attention was elsewhere."

He goes on to say:

> "While listening to their program execute, subjects took advantage of their unburdened visual processing abilities to manipulate the graphical interface in order to gain better perspectives on their programs".

This work shows that sound at the interface can help by unburdening users' visual systems so that they can perform other tasks simultaneously. It also allows them to perform better as they can monitor more data from their programs because of more sensory channels.

Other workers in the field have developed systems to display graphs and various types of numerical data in sound (see [155], [113] and [27]). The rest of the sections in this chapter will describe, in detail, some of the most important work in the area of auditory interfaces.

3.7 SOUNDTRACK

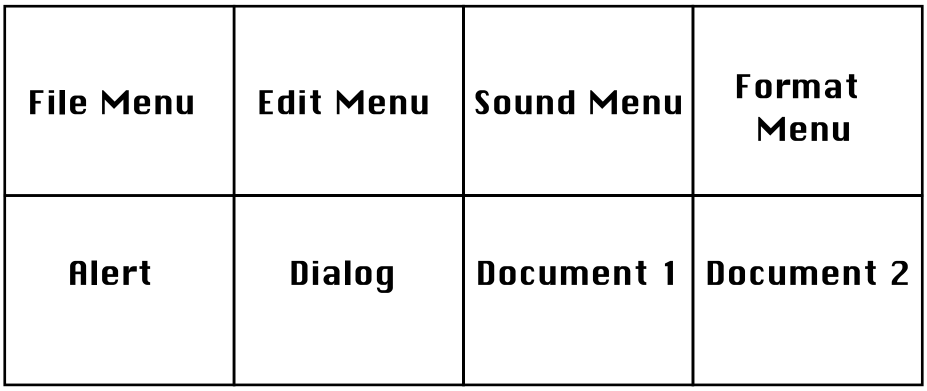
This is a word-processor designed to be used by blind persons and was developed by Edwards [62, 63]. It uses tones and synthetic speech as output. It is designed so that the objects a sighted user would see in an interface, for example menus and dialogues, are replaced by auditory equivalents that are analogies of their visual counterparts.

3.7.1 Soundtrack's main interface

The interface is constructed from auditory objects - these are objects the user can interact with. They are defined by a location, a name, a tone and an action. They are arranged into a grid of two layers (see Figure 3.1), at this level they are analogous to menus.

Each auditory object made a tone when the cursor entered it and these tones could be used to rapidly navigate around the screen. Soundtrack used sinewaves for its audio feedback. Chords were built-up for each menu dependent on the number of menu items. In the case of the alert menu a single sinewave was played as there were no items in the menu (this menu just displayed error messages if they occurred). For the edit menu (Figure 3.2) a chord of four notes was played because there were four menu items.

The base tones increased in pitch from left to right - as in the normal representation of a musical scale (for example on a piano) and the top layer used higher pitches than the bottom. Using these two pieces of information a user could quickly find his/her position on the screen. If any edge of the screen was reached a warning sound was played. If at any point the user got lost or needed more precise information he/she could click on an object and it would speak its name.

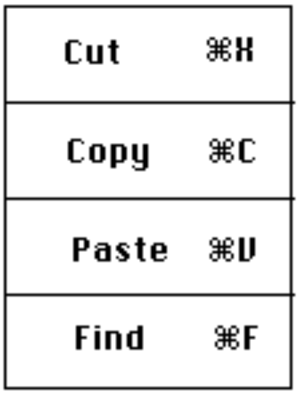


File Menu	Edit Menu	Sound Menu	Format Menu
Alert	Dialog	Document 1	Document 2

Figure 3.1: *Soundtrack's main screen (from [63]).*

3.7.2 Soundtrack's sub-menus

Double-clicking on a menu object takes the user to the second level of Soundtrack - the menu items associated with the chosen menu. In Figure 3.2 the edit sub-menu is shown. Moving around is similar to moving on the main screen: Each item has a tone and when clicked on will speak its name. When moving down the menu the pitch of the tones decrease and when moving up the pitches increase. Double-clicking on an item causes its action to take place. Some menu items also have key equivalents and so can be accessed directly from the keyboard at the top level.



Cut	⌘X
Copy	⌘C
Paste	⌘V
Find	⌘F

Figure 3.2: *Soundtrack's Edit Menu (from* [63]*).*

Some of Soundtrack's basic functions will now be described. The edit menu allows basic editing operations and a find facility. The file menu allows creation of new files, the opening and saving of existing files and quitting. The sound menu allows the user to customise the sounds that Soundtrack produces. The format menu allows, for example, the emboldening or underlining of text. Double-clicking on the alert item shows any alerts or error messages that need to be displayed. The dialogue menu is used when the system needs more information from the user. For example, if the user selects Quit before saving a document, Soundtrack must ask if they want to save the document or not.

Results from Edwards [63] show that most of the test subjects counted the number of tones they heard to find their position within the interface, they did not use perception of the changes in pitch. However, one user who had musical training did use the pitches. It may have been that as there were only a few tones counting was easier, if there had been more then counting would have become too slow and pitch perception would perhaps have been used.

The main drawback of Soundtrack was that it was not a general solution to the problem of visual interfaces, it could only be applied to a word-processor, the same solution could not be used to help guide a user around the Macintosh desktop, for example. Soundtrack did prove, however, that a full auditory representation of a visual interface could be created with all the interactions based in sound. It showed that the human auditory system was capable of controlling an interaction with a computer and therefore using auditory interfaces was possible.

3.8 AUDITORY ICONS

Gaver [73, 74] has developed the idea of auditory icons. These are natural, everyday sounds which are used to represent actions and objects within an interface. The SonicFinder was developed from these ideas. This was one of the first attempts to actually integrate non-speech audio into a human-computer interface.

3.8.1 Environmental sounds

Mountford & Gaver [119] suggest that sound can provide information about many different things within the environment, for example:

- ❖ *Physical events:* Whether a dropped glass broke or bounced.
- ❖ *Invisible structures:* Tapping on a wall to find if there is a hollow behind it.
- ❖ *Dynamic changes:* As a glass is filled with liquid a listener can hear when it is full.
- ❖ *Abnormal structures:* A malfunctioning engine sounds different to a normal one.
- ❖ *Events in space:* The sound of footsteps indicate that someone is approaching.

Gaver uses sounds of events that are recorded from the natural environment, for example tapping or smashing sounds. His research is based on the ideas of Vanderveer [169] and Warren & Verbrugge [173]. This work suggests that people do not listen to the pitch and timbre of sounds but to the sources that created them. When pouring liquid a listener hears the fullness of the receptacle, not the increases in pitch. Warren & Verbrugge suggest that (p 705) “Identification of sound sources, and the behaviour of those sources, is the primary task of the auditory system”. Vanderveer performed tests where subjects had to identify various common sounds, for example tearing paper or jingling keys. She found that subjects tended to be able to match sounds to sources very accurately, up to 95% in some cases. Confusions tended to occur when there was more than one possible source, for example walking was confused with hammering or filing with scratching, but never walking with filing. Another important property of everyday sounds is that they can convey multidimensional data. When a door slams a listener may hear: The size and material of the door; the force that was used; and the size of room on which it was slammed. This could be used within an interface so that selection of an object makes a tapping sound, the type of material could represent the type of object and the size of the tapped object could represent the size of the object within the interface. Björk [20] and Ballas & Howard [13] have also done work in this area.

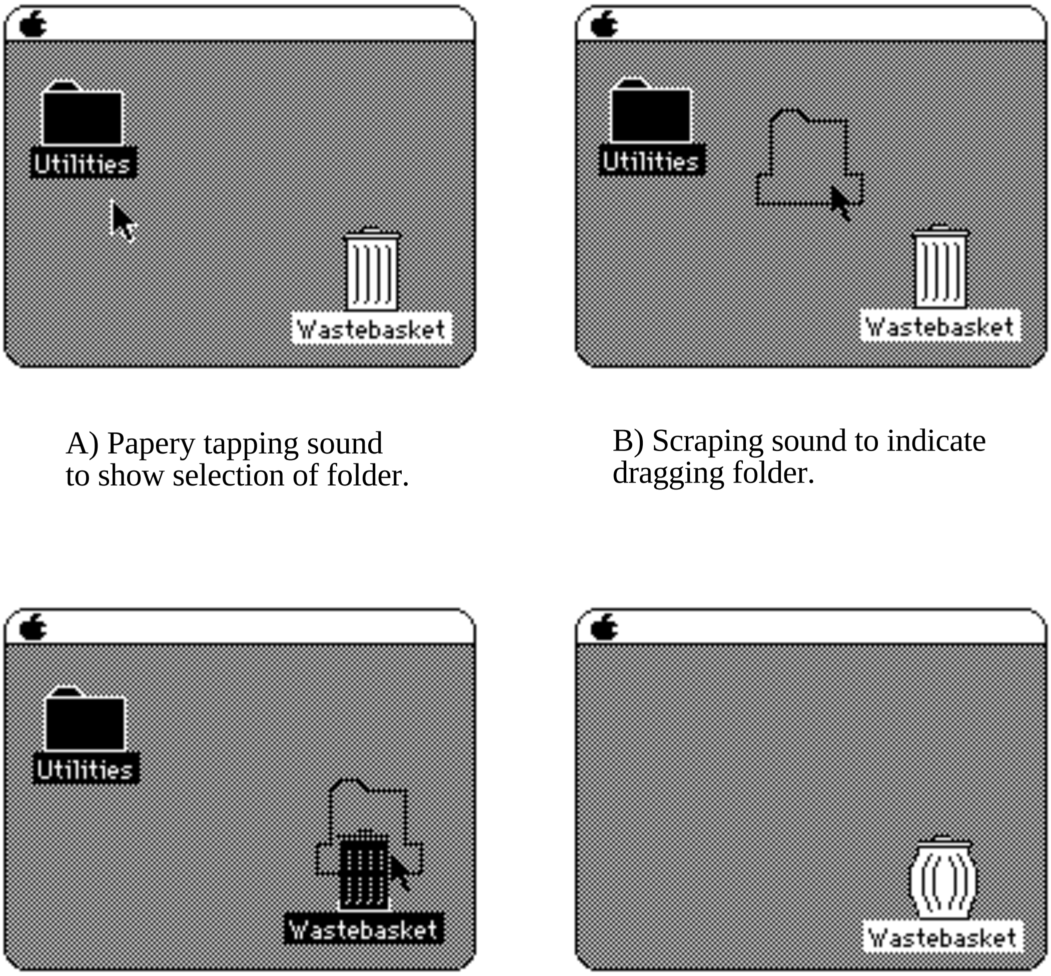
3.8.2 The SonicFinder

Gaver used these ideas to create auditory icons and from these built the SonicFinder [74]. This is an interface that runs on the Apple Macintosh alongside the ordinary Finder and provides auditory representations of some objects and actions within the interface. This system is not designed for blind users but as an aid for sighted Mac users. Files are given a wooden sound, applications a metal sound and folders a ‘papery’ sound. The larger the object the deeper the sound it makes. Thus, selecting an application means tapping it - it will make a metal sound which will confirm that it is an application and the deepness of the sound will indicate its size. Copying uses the idea of pouring liquid into a receptacle. The rising of the pitch indicates that the receptacle is getting fuller and when it is full the copy is complete. To delete a file on the Macintosh it is dragged and dropped into the wastebasket. This makes the sound of smashing dishes to indicate destruction.

To demonstrate how the SonicFinder works a simple interaction is provided in Figure 3.3 showing the deletion of a folder. In A) a folder is selected by tapping on it, this causes a ‘papery’ sound indicating a folder. In B) the folder is dragged towards the wastebasket causing a scraping sound. In C) the wastebasket becomes highlighted and a ‘clinking’ sound occurs when the pointer reaches it. Finally, in D), the folder is dropped into the wastebasket and a smashing sound occurs to indicate it has been deleted (the wastebasket becomes ‘fat’ to indicate there is something in it).

The SonicFinder mainly adds redundant information to the Macintosh interface. Most of the actions and objects that have sounds have graphical representations as well. It may be that some parts of the interface could be represented better purely in sound with no need for a graphical representation to back them up. The research discussed in this thesis suggests that sound has a useful place on its own in the interface, not just as redundant feedback. Problems can occur with representational systems such as auditory icons because some abstract interface actions and objects have no obvious representation in everyday sound. Gaver used a pouring sound to indicate copying because there is no natural equivalent; this is more like a ‘sound effect’. He suggests the use of movie-like sound effects to create sounds for things with no easy representation. This may cause problems if the sounds are not chosen correctly as they will become more abstract than representational and the advantages of auditory icons will be lost.

Gaver deals with the two questions that this thesis is concerned with very simply, he suggests natural sounds should be used and that they should be used in the ways suggested by the natural environment. As mentioned, this is not always effective because there are many situations in an interface that have no natural equivalent.



A) Papery tapping sound to show selection of folder.

B) Scraping sound to indicate dragging folder.

C) Clinking sound to show wastebasket selected

D) Smashing sound to indicate folder deleted.

***Figure 3.3**: An interaction showing the deletion of a folder in the SonicFinder (from* [74]*).*

3.8.3 SharedARK

Gaver extended his ideas of auditory icons into the area of large-scale collaborative environments [78]. Gaver & Smith based their work around the Shared Alternative Reality Kit (SharedARK). In this system a virtual physics laboratory was modelled. Multiple users could perform ‘virtual’ experiments on “objects in an environment that extends far beyond the view offered by their screens” (p 735). Three groups of sounds were used: Confirmatory sounds, process and state sounds and navigation aids.

The confirmatory group of sounds used many of the principles from the SonicFinder. Clicking on a button made a tapping sound (the same as for selecting a file or folder in the SonicFinder) or putting one object on top of another made a wooden sound (similar to moving an icon over a folder or the wastebasket, where a clunk is made to show the target has been hit). These sounds are again redundant as the system provides visual feedback for actions, but Gaver & Smith suggest (p 737) “…these sounds seem to provide feedback about actions in a way that is immediate, intuitive and engrossing”.

The process and state information sounds used in SharedArk have two levels. There are global states for the whole system and the sounds representing these can be heard everywhere within the system, for example the user can experiment with the laws of motion; when these laws are turned on a sound will be played that can be heard anywhere in SharedARK. These sounds are designed to fade into the background of consciousness. Other more localised sounds are then used for each of the specific experiments involving the laws of motion. If the user moves from one experiment to another, the new experiment’s sound will be louder than that of the other, which is further away. Wherever the user moves, the sounds of the global state of the system will be the same. This type of constant audio feedback is an important step forward from previous systems that just presented event information in sound. For example, beeps when an error occurs or when the end of a file is reached. Much of the feedback displayed in graphical interfaces is constant, changing very little over time. Constant audio feedback is a useful extension to simple event feedback, as will be described in Chapter 7.

One of the most interesting concepts in SharedARK is the ‘soundholder’. These are auditory landmarks that users can navigate by. They can be placed anywhere in the system and constantly emit sounds whose volume decreases as the user moves away from them or increases as the user gets closer (similar to the ideas described by Pitt & Edwards above). They can be located near areas of work so that users can move around a space much bigger than the screen without getting lost and easily return to where they were by moving towards the soundholder. Gaver & Smith suggest using environmental sounds such as bird calls or burbling streams for these, as they are very distinct and easy to remember.

3.8.4 ARKola

The ARKola system (Gaver, Smith & O'Shea [79]) again used the SharedARK but this time a soft drinks factory was modelled. The simulation consisted of a set of nine machines. The machines were split into two groups: Those for input and those for output. The input machines supplied the raw materials, the output machines capped the bottles and sent them for shipping and financing. Each machine had an on/off switch and a rate control. The aim of the simulation was to run the plant as efficiently as possible, avoid waste of raw materials and make a profit by shipping bottles. Two subjects used the system at the same time from physically separate locations. They communicated via a two-way audio/video link.

Each of the machines had a sound to indicate its status over time, for example the bottle dispenser made the sound of clinking bottles. The rhythm of the sounds reflected the rate at which the machine was running. If a machine ran out of supplies or broke down its sound stopped. Sounds were also added to indicate that materials were being wasted. A splashing sound indicated that liquid was being spilled, the sound of smashing bottles indicated that bottles were being lost. The system was designed so that up to fourteen different sounds could be played at once. To reduce the chance that all sounds would be playing simultaneously, sounds were pulsed once a second rather than playing continuously.

Gaver *et al.* suggest that users could effectively monitor the status of on-going processes via the auditory cues. The subjects used the rhythmic information from the machines to check they were working correctly and also listened to the combined sounds of the whole plant to hear if all the machines were okay. Problems did occur with some of the warning sounds used as Gaver *et al.* indicate (p 89):

> "the breaking bottle sound was so compelling semantically and acoustically that partners sometimes rushed to stop the sound without understanding its underlying cause or at the expense of ignoring more serious problems."

Another problem was that when a machine ran out of raw materials its sound just stopped, users sometimes missed this and did not notice that the sound had stopped.

Gaver *et al*. say (p 89) "traditional uses of sound indicate that *something* is happening, but not *what* is happening. Auditory icons seem admirably well suited for conveying semantic information on events." In their results, Gaver *et al.* say that subjects did not forget the meanings of the sounds. One of the biggest advantages of auditory icons is the ability to communicate meanings which listeners can easily learn and remember, other systems (for example earcons, see the next section) use abstract sounds where the meanings are harder to learn. The disadvantage of auditory icons is that, according to Blattner, Sumikawa & Greenberg [25], having a large number of them creates the problem of memorising each one as a distinct entity because there is no structure linking them together. For example, there may be no common sound to represent errors, so that errors which are similar may have very different, unrelated sounds. There is also the problem, as mentioned earlier, of creating an auditory icon to represent something which does not have a natural sound (for example a window), or where the natural sound does not describe the action within the interface (for example opening or closing a window). This is not a problem for more abstract systems where any sound can be used, the listener must learn the mapping between it and the meaning.

3.8.5 The RAVE System and ShareMon

The RAVE system [77] developed at EuroPARC was designed to (p 27) "allow physically separated colleagues to work together effectively and naturally". The system was designed to enable collaboration between workers at EuroPARC. One example of this was the installation of two-way video cameras into offices so that people could communicate whilst still being physically separated. To deal with some of the issues of privacy this brought up, auditory icons were added. For example, when someone started to look into a particular office, a door opening sound was played to the people in that office so that they knew they were being looked at. When the connection was broken a door closing sound was played. Other sounds included: A knock or telephone bell to indicate a videophone request and footsteps to indicate sweeps (where one person briefly sweeps their video 'eyes' over all the other offices to see who is around - similar to walking by and looking through the doors). As Gaver says (p 31): "The auditory cues provide information about what kind of connection is being made, over and above information about the existence of a connection alone". These sounds provide a powerful way of communicating information to users without distracting their attention.

ShareMon [42, 43] uses auditory icons to give information about background file sharing activity on the Apple Macintosh. Cohen wanted to present this information without disrupting foreground activities. ShareMon indicates when another user logs-on to your machine in a similar way to RAVE above: A knocking sound is given when a user logs-on and a door slamming sound when they log-off. Other sounds include: A draw opening/closing sound to indicate open/close file; a chair creak to remind the user that there is still a connection in place; and walking sounds to indicate level of copying activity (walking = low activity, running = high activity). He reports one interesting problem with the sounds he used ([43], p 64):

> "For example one user told me this story when she heard knocking sounds followed by walking sounds [log-on then low activity]: "Right after you're knocking you're walking -

> you're outside knocking on the door and my computer is not responding. So you're walking back and forth. It's like you're a father expecting a baby". Unfortunately this wasn't the intended story. The sounds were supposed to mean that a guest had logged on and started copying a file".

This indicates one potential problem with auditory icons. As Deutsch [51] suggests that when naturally occurring sound are presented in isolation they can become ambiguous. In the everyday world we overcome this ambiguity by using information from other sounds or senses. Many other things may accompany the presentation of a sound. There might be some visual event associated with it, for example a bang and puff of smoke from a gun. A sound will be further modified by the environment it passes through before it reaches the ear, for example by reverberation or damping due to the surfaces it comes into contact with. If this contextual information is missing then what might be an intuitive sound in natural circumstances can become ambiguous.

The meaning of auditory icons is supposed to be intuitive but, as shown above, they may mean different things to different people. Therefore their meanings must be learned and then they lose one of their advantages over more abstract systems (such as *earcons* described in the next section) where the sound → action mapping must be learned explicitly. To try and avoid such problems great care must be taken, as Cohen says (p 64): "In my experience with ShareMon, I've found that it's a difficult task constructing sounds which tell the right story and are also pleasant and emotionally neutral".

As discussed above, Gaver's answer to the two questions in this thesis may not always work. His method of finding where to use sound can break down in an abstract situation such as file sharing. Cohen looked at different types of auditory icons to see which were the most effective. Some of the sounds he used were not suggested by the natural environment. For example, a chair creaking has nothing to do with maintaining a connection for sharing files. His use of sounds is more abstract. The sounds are still natural but they are not used in the ways suggested by the natural environment.

3.8.6 Parameterising auditory icons

Gaver [75, 76] is working on a system of parameterising [sic] auditory icons. He says ([76], p 228):

> "Auditory icons not only reflect categories of events and objects as visual icons do, but are *parameterised* to reflect their relevant dimensions as well. That is, if a file is large, it sounds large. If it is dragged over a new surface, we hear that new surface. And if an ongoing process starts running more quickly, it sounds quicker".

If auditory icons are to be generated in real-time, rather than just be stored as fixed sound samples, then ways must be found of controlling their parameters and synthesising them directly. This would allow families of auditory icons to be created which varied along certain attributes (giving auditory icons some of the advantages claimed for the system described in the next section). Gaver puts forward some early work to solve this problem. He has developed algorithms to allow the description of sound properties. For impact sounds he can define the hardness of the hammer hitting an object, the material the object is made from and its size. He can define whether an object sounds like it is bouncing, breaking or spilling and also scraping and dragging. He has used FM synthesis techniques [41] to simulate machine sounds and can control the size of the machine and its speed of operation.

This work is in its early stages but if it continues may allow auditory icons to become very much more flexible than they are now. As the above sections have shown, auditory icons have been used in many different situations and interfaces. They have been shown to be very effective at communicating information although there can be problems with the intuitive mappings. The next section describes the main alternative to Gaver's system.

3.9 EARCONS

Earcons were developed by Blattner, Sumikawa & Greenberg [25], Sumikawa, Blattner, Joy & Greenberg [163], Sumikawa, Blattner & Greenberg [162] and Sumikawa [164]. They use abstract, synthetic tones in structured combinations to create auditory messages. Blattner *et al.* ([25], p 13) define earcons as "non-verbal audio messages that are used in the computer/user interface to provide information to the user about some computer object, operation or interaction". Unlike Gaver's auditory icons, there is no intuitive link between the sound and what it represents; the link must be learned by the listener. Earcons use a more musical approach than auditory icons. The following sections describe the construction of earcons from smaller units called *motives* along with descriptions of two different types of earcons. Some issues in learning and remembering earcons are discussed and then some systems that use earcons are given.

3.9.1 Motives

Earcons are constructed from simple building blocks called motives. These are short, rhythmic sequences of pitches that can be combined in different ways. Sumikawa *et al.* ([162], p 5) describe them thus:

> "A motive is a brief succession of pitches arranged in such a way as to produce a tonal pattern sufficiently distinct to allow it to function as an individual recognisable entity".

They go on to say:

> "The eloquence of motives lies in their ability to be combined to create larger recognisable structures. The repetition of motives, either exact or varied, or the linking of several different motives produces larger, more self sufficient patterns. We use these larger structures for earcons".

The most important features of motives are:

- *Rhythm*: As mentioned in the previous chapter, this is one the most important characteristics of a sound [49]. Changing the rhythm of a motive can make it sound very different. Blattner *et al.* [25] describe this as the most prominent characteristic of a motive.

- *Pitch*: There are 96 different pitches in the western musical system and these could be combined to produce a large number of different motives. Sumikawa *et al.* suggest that random combinations of pitches should not be used, but that they should be taken from one octave for easier manipulation.

- *Timbre*: Motives can be made to sound different by the use of different timbres, for example playing one motive with the sound of a violin and the other with the sound of a piano. Sine waves can be used to provide ‘colourless’ sounds (i.e. sounds with no distinct timbre).

- *Register*: This is the position of the motive in the musical scale. A high register means a high pitched note and a low register a low note. The same motive in a different register can convey a different meaning.

- *Dynamics*: This is the volume of the motive. It can be made to increase as the motive plays (crescendo) or decrease (decrescendo). A crescendo could be used to give the idea of zooming a window, for example.

Sumikawa [164] defines rhythm and pitch as the *fixed* parameters of earcons and timbre, register and dynamics as the *variable* parameters. The fixed parameters are what define a motive, the variable parameters change it. Sumikawa put forward some general guidelines for use in the creation of motives. She gave general rules to follow but few precise instructions. In order to reduce the possible number of combinations of the above parameters, Sumikawa suggested some restrictions. Only seven time divisions should be used when creating rhythms and notes should be kept within a range of eight octaves of twelve notes. Semitone gaps should be avoided as they can create incorrect melodic implications; earcons should be musically neutral. She suggested that only four timbres should be used (sine wave, square wave, sawtooth wave and triangular wave) with three registers (low, medium and high) and a total of five dynamics (soft, medium, loud, soft to loud and loud to soft). Sumikawa said that motives should be no longer than three or four notes or they will become too long for the user to easily remember and may also take too much time to play when in combination. She give little empirical evidence to support many of these restrictions of the parameters. Work described later in this thesis attempts to put forward a more detailed set of guidelines based on experimental results.

3.9.2 Earcon construction

As mentioned above, earcons are constructed from motives. Sumikawa ([164], p 64) suggested some principles to keep in mind when constructing an earcon: “It should convey one basic meaning, be brief, simple and distinct from other earcons, and be easy to remember, identify and understand”. Unfortunately she gave only general guidelines for creating earcons which means that following the above rules is difficult. She suggested three ways in which motives could be manipulated to create earcons:

- *Repetition*: Exact restatement of a preceding motive and its parameters.
- *Variation*: Altering one or more of the variable parameters from the preceding motive.
- *Contrast*: A decided difference in the pitch and/or rhythmic content from the preceding motive.

These manipulations can be used in different ways. Blattner *et al*. [25] describe two types of earcons based on them: Compound earcons and family, or hierarchical, earcons.

Compound earcons

Compound earcons are the simplest of the two types. Compound earcons could be used to represent the actions and objects which make up an interface. They could then be combined in different ways to provide information about any interaction in the interface. If a set of simple, one element motives was created to represent various system elements, for example ‘create’, ‘destroy, ‘file’ and ‘string’ (see Figure 3.4) these could then be concatenated to form earcons. In the figure the earcon for ‘create’ is a high pitched sound which gets louder, for ‘destroy’ it is a low pitched sound which gets quieter. For ‘file’ there are two long notes which fall in pitch and for ‘string’ two short notes that rise in pitch. In Figure 3.5 the compound earcons can be seen. For the ‘create file’ earcon the ‘create’ motive is simply followed by the ‘file’ motive. This provides a simple and effective method for building up complex messages in sound.

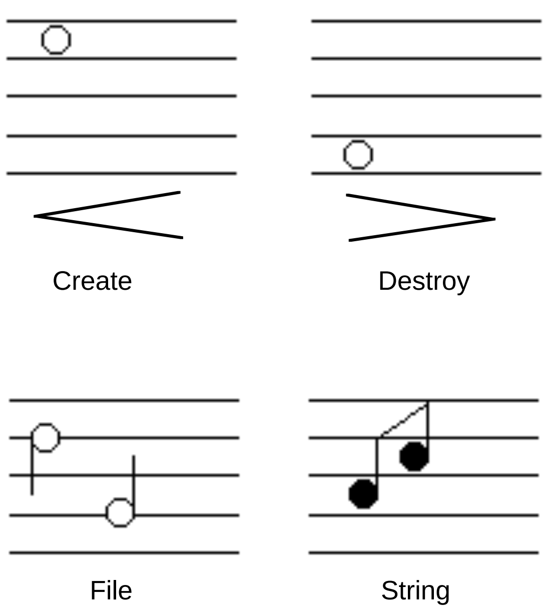


Figure 3.4*: The four audio elements ‘create’, ‘destroy’, ‘file’ and ‘string’ (from* [25]*).*

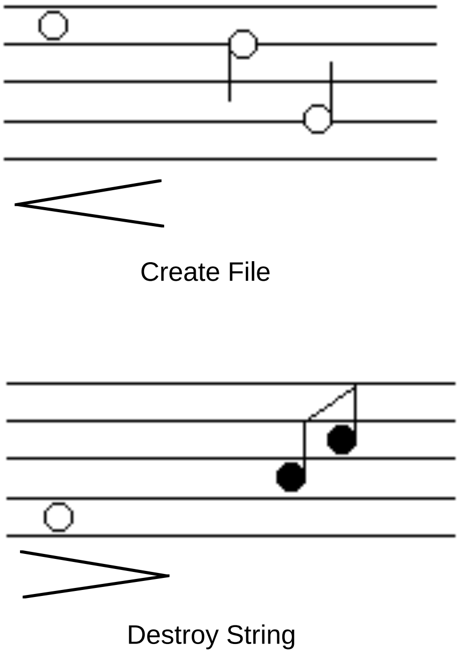


Figure 3.5*: Combining audio elements ‘create file’ and ‘destroy string’ (from* [25]*).*

Hierarchical earcons

The second type of audio messages, called family or hierarchical earcons, provide a more powerful, hierarchical system. Each earcon is a node on a tree and inherits all the properties of the earcons above it. Figure 3.6 shows a hierarchy of family earcons. There is a maximum of five levels to the tree as there are only five parameters that can be varied (rhythm, pitch, timbre, register and dynamics). In the diagram the top level of the tree is the family rhythm, in this case it is a sound representing error. This sound just has a rhythm and no pitch, the sounds used are clicks. The rhythmic structure of level one is inherited by level two but this time a second motive is added where pitches are put to the rhythm. At this level, Sumikawa suggests the timbre should be a sine wave, which produces a ‘colourless’ sound. This is done so that at level three the timbre can be varied. At level three the pitch is also raised by a semitone to make it easier to differentiate from the pitches inherited from level two. Other levels can be created where register and dynamics are varied.

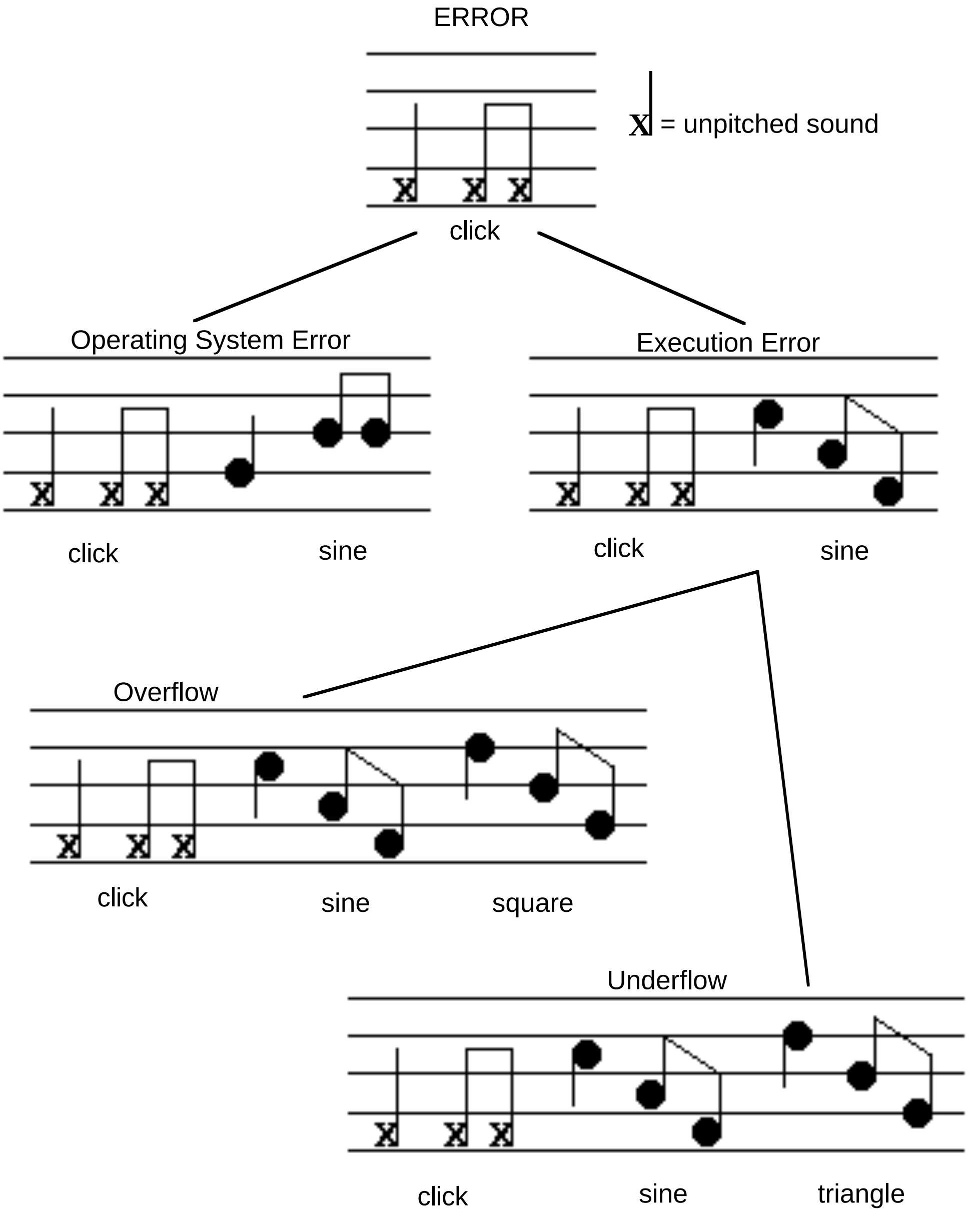


Figure 3.6: *A hierarchy of family earcons representing errors (from* [25]*).*

Blattner *et al*. suggest that for novice users to recognise the sounds, the full earcon at the leaf node of the tree may need to be played (for example, all three motives of the underflow error). Expert users should be able to recognise the earcon if only the last motive was played (just the part labelled triangle in the underflow error). This thesis suggests that, in order for earcons to be useful at the interface, they must be able to keep pace with the interactions going on; if the whole three-motive earcon must be played then they will not be able to keep up. The experiments described in this thesis test hierarchical earcons with just the last motive played to discover their effectiveness.

3.9.3 Learning and remembering earcons

An important factor in the usability of earcons is memorability. If they become too long this may be a problem so Sumikawa suggests they should be kept as short as is necessary to get their message across. Deutsch [49] suggests that structured sequences of tones should be more easily remembered than random tones. The strong rhythmic structure will also aid memory. With the inheritance hierarchy there is only one new piece of information to remember at each level, thus making family earcons easier to remember. Initially, there will be much to learn but later extensions will be simple. For example, if another operating system error was to be added to Figure 3.6 only one new motive would have to be learned in the whole earcon. In systems such as Gaver's auditory icons a whole new sound may have to be learned which is unrelated to any other.

Family earcons seem to be a powerful system for representing hierarchical structures such as menus or errors. When navigating through menus the user would normally only be able to see the menu that they were on. Family earcons could provide auditory information about where the user was in the hierarchy without adding to the visual information present. Unfortunately, Blattner *et al*. did not implement a system of earcons and perform experiments to see how effective they were: Their usefulness is unknown.

The use of the characteristics of sound mentioned above, for example rhythm, pitch, dynamics etc., has a strong psychoacoustic basis. Earcons are based around different rhythms and this is one of the most important methods for grouping sounds into sources (see Chapter 2 on auditory pattern recognition). Sumikawa suggests keeping the notes used from the same octave and in the same scale. This fits in well with the work of Dewar, Cuddy & Mewhort [53] who suggested that listeners can better detect differences between groups of sounds if all the notes in one sound are in a different scale to the other. Keeping all the notes in one octave also minimises pitch perception problems where a listener can mistake the octave to which a note belongs [52]. Loudness perception also plays its part as different loudnesses are used to differentiate earcons. Using the ideas of psychoacoustics two further sound variations could be used to help differentiate earcons from one another. According to the work on auditory pattern recognition, coherence could be used to help group the components of an earcon into a sound source. Modulation could be applied to all the motives of an earcon to cause the separate sounds to be grouped more concretely. Spatial positioning could also be used. Using localisation information to position earcons in different locations in space could also aid grouping of sounds into separate sources (see Moore [118] pp 239-241 for more details).

Earcons have some advantages over auditory icons because they have a strong structure to link them together. This may reduce the learning time and the memory load, but as they have not yet been tested this is unknown. Auditory icons may be more easily recognisable as they are based on natural sounds (which human beings have evolved to listen to over a long period of time) and the sounds contain a semantic link to the objects they represent. Earcons are completely abstract: The sound has no relation to the object that it represents. This may make their association to actions or objects within an interface more difficult. Again, as they have not been tested this is unknown. Cohen [43] pointed out some problems, described above, in that people's intuitive mappings for a sound may be different, and so the meaning of an auditory icon would have to be learned (like an earcon). Problems of ambiguity can occur when natural sounds are taken out of the natural environment: Other cues to help a listener recognise them are lost. If the meanings of auditory icons must be learned then they lose some of their advantages. Earcons have a built-in structure that can easily be manipulated (for example changing pitch or intensity). Auditory icons are only just beginning to get this [76] so that building structures with them at present is difficult. An important step will be to test a system of earcons to discover their advantages and disadvantages. A major part of this thesis is to do just this and find out how effective earcons really are.

Blattner *et al.* [25] answer one of the questions this thesis is concerned with. They suggest what sounds should be used (earcons) but they do not make any suggestions about where they should be used in the interface. This is investigated as part of the work in this thesis.

3.9.4 Auditory maps and other examples of earcons

There are very few examples of systems using earcons. At the start of the research described in this thesis there were none. The example systems that do currently exist are described in the following section. Blattner, Papp & Glinert [26] added earcons to two-dimensional maps. They used sound (p 459):

> "…because the addition of visual data requires that space be allocated for it, [and] a saturation point will eventually be reached beyond which interference with text and graphics already on the screen cancels out any possible benefit".

Blattner *et al.* used a map of floor plans of the Lawrence Livermore National Laboratory. To this they added information in sound such as type of computer equipment contained in the building, security clearance required and jobs of those in the building. Hierarchical earcons were used for the sounds. For example, a three note saxophone earcon indicated an administrative building and a tom-tom represented the security restriction. The faster the tom-tom was played the higher the restriction. The user could click on a building to hear what it contained or they could use an area selector. Any building which was within the area selector would play its earcons, concurrently with the other buildings. This technique allowed much more data to be represented than in the graphical case. Although no experimental testing was reported by Blattner *et al.* the system seemed to have much potential. A similar approach was also taken by Kramer [104].

Barfield, Rosenberg & Levasseur [14] carried out experiments where they used earcons to aid navigation through a menu hierarchy. They say (p 102): "…the following study was done to determine if using sound to represent depth within the menu structure would assist users in recalling the level of a particular menu item". The earcons they used were very simple, just decreasing in pitch as a subject moved down the hierarchy. The sounds lasted half a second. They describe them thus (p 104):

> "…the tones were played with a harpsichord sound and started in the fifth octave of E corresponding to the main or top level of the menu and descended through B of the fourth octave".

These sounds did not fully exploit all the advantages offered by earcons (for example, they did not use rhythm or different timbres) and they did not improve user performance in the task. If better earcons were used then a performance increase may have been found. This work shows that there is a need for a set of guidelines for the creation of effective earcons. If earcons are created without care then they may be ineffective. Later in this thesis an investigation of earcons is undertaken and a set of guidelines are produced to help in such cases as this. One other reason for the lack of success of the earcons in this experiment was that they might not have been used in the best place in the interface. There are no rules for where to use sounds, the choice is up to individual designers and so mistakes can occur. Later in the thesis an investigation is undertaken of where sound can effectively be used in an interface.

Blattner, Greenberg & Kamegai [24] discuss the use of earcons to present specific information such as speed, temperature or energy, in a system to sonify turbulent liquids. They suggest that earcons could be used to present this information so that the user would not have to take their eyes off the main graphical display. Unfortunately, the system they discuss was never implemented.

Some recent work by Stevens, Brewster, Wright & Edwards [160] used earcons to provide a method for blind readers to 'glance' at algebra. They say that a glance, or overview, is very important for planning the reading process. There is currently no way to do this. They suggest using earcons. The parameters described above for manipulating earcons are very similar to those describing prosody in speech. Stevens *et* *al.* combined algebra syntax, algebraic prosody and earcons to produce a system of *algebra earcons*. They describe a set of rules that can be used to construct algebra earcons from an algebra expression.

Different items within an algebra expression are replaced by sounds with different timbres, such as: Piano for base-level operands, violin for superscripts and drums for equals. Rhythm, pitch and intensity are then defined according to other rules. Stevens *et al.* experimentally tested the earcons to see if listeners could extract algebraic structure from the sounds and identify the expressions they represented. Their results showed that subjects performed significantly better than chance. This work again shows earcons are a flexible and useful method of presenting complex information at the interface.

3.10 TWO COMPARISONS OF EARCONS AND AUDITORY ICONS

3.10.1 Comparison 1: Jones & Furner

There have been two main comparisons of earcons and auditory icons. The first was carried out, by Jones & Furner [100]. They performed two experiments. In the first they tried to find out which types of sounds listeners preferred. Subjects were given typical interface commands (for example, delete or copy) and were played a sample earcon, an auditory icon and some synthesised speech. In the experiment, earcons were preferred to auditory icons. A second experiment was carried out to see if subjects could associate sounds with commands. The subjects were played a sound and had to match it to a command in a list. This time auditory icons proved to be more popular than earcons. This may have been because family earcons would initially be harder to associate to commands (as they have no inherent meaning and take time to learn) whereas auditory icons have a semantic relationship with the function they represent.

In both experiments the subjects scored highest with the synthesised speech which is not surprising as speech contains all the information necessary and does not require any learning. The paper does not describe the nature of the earcons and auditory icons used. It may have been that the cues were not well designed which would have affected preference and association. The reasons why speech is not being used were outlined at the beginning of this section and these still hold true as speech loses some of its advantages when used in different situations.

In the preference experiment a further set of abstract commands was presented to the subjects. There was no significant difference between earcons and synthetic speech in this case but the preference for auditory icons was much lower. This matches expectations as there is no semantic link between an abstract auditory icon and its meaning. It is interesting to note that there was no difference in preference between the earcons and synthetic speech in this case.

3.10.2 Comparison 2: Lucas

A further comparison experiment was undertaken by Lucas [111]. He conducted a more detailed analysis of earcons, auditory icons and synthetic speech. He presented subjects with sound stimuli of the three types and they had to choose which command they felt most closely fitted the sound. He conducted a second trial a week after the first.

His results showed that there was no difference in response time between earcons and auditory icons. Speech was significantly faster. There were no differences in response between trial one and two. However, subjects did make fewer errors on trial two. This would indicate that with more training auditory cues could easily be learned. There were no significant differences in error rates between earcons and auditory icons. There were no errors in the speech condition (as the stimuli were self-explanatory). After the first trial, half of the subjects were given an explanation of the sounds used and the design methods. These subjects showed a decrease in error rates on trial two. Lucas (p 7) says : “This indicates that a knowledge of the auditory cue design can improve the accuracy of cue recognition…”.

3.10.3 Discussion of the comparisons

These two comparisons have shown little difference between earcons and auditory icons. It may be that each has advantages over the other in certain circumstances and that a combination of both might be best. In some situations the intuitive nature of auditory icons may make them favourable. In other situations earcons might be best because of the powerful structure they contain, especially if there is no real-world equivalent of what the sounds are representing. Indeed, there may be some middle ground where the natural sounds of auditory icons can be manipulated to give the structure of earcons. Cohen [44] proposes that there is a continuum of sound from the literal everyday sounds of auditory icons to the abstract sounds of earcons. Objects or actions within an interface that do not have an auditory equivalent must have an auditory icon made for them. This then has no semantic link to what it represents: Its meaning must be learned. The auditory icon then moves more towards the abstract end of the continuum. When hearing an earcon, the listener may hear and recognise a piano timbre, rhythm and pitch structure as a kind of ‘catch-phrase’; he/she will not hear all the separate parts of the earcon and work out the meaning from them. The earcon will be heard more as a whole source and thus the perception of the earcon moves more towards the representational side of the continuum. Therefore, earcons and icons are not necessarily as far apart as they might appear. Both have advantages and disadvantages but these can be maximised/minimised by looking at the properties of each. This thesis will attempt to look more closely at the properties of earcons because, as yet, little is known about them.

One drawback of earcons is that it is unclear as to how effective they are. They have not been tested in any implementations (as auditory icons have) so it is not known whether listeners will be able to extract the complex information from them. It is also clear from the experiments of Barfield *et al.* [14] that creating earcons is not a simple matter; it is easy to create ineffective ones. Therefore some clear experiments to test the usability of earcons are needed as is a set of guidelines to help designers build effective ones. The work described in this thesis attempts to deal with these problems. Earcons are experimentally tested and from the results a set guidelines are produced to help designers of earcons.

In the work on earcons and auditory icons little indication is given as to where sounds should be used in the interface. Gaver suggests that they should be used in ways suggested by the natural environment but in a computer interface this is not always possible. One important step forward would be to have a method to find where sound in the interface would be useful. This thesis proposes such a method.

3.11 AUDIO WINDOWS

Ludwig, Pincever & Cohen [112], Cohen & Ludwig [45] and Cohen [44] have done work in adding sound to graphical window systems using the ideas of Wenzel *et al*. [179] mentioned in the previous chapter. They use digital signal processing techniques to control multiple sound sources. Ludwig *et al*. suggest that various sound effects can be applied to sounds to distinguish them from each other:

- ❖ *Self-animation*: Frequency-dependent phase distortion can be used to ‘excite’ the sound. This makes the sound more noticeable by accentuating frequency variations. This can also include adding distortion.
- ❖ *Thickening*: (also called ‘doubling’) produces a thicker sound by chorusing or pitch-shifting the signals so that the sound plays at several different pitches.
- ❖ *Distancing*: By using echo and reverberation the source can be made to sound further away.
- ❖ *Muffling and thinning*: By using low-pass and high-pass filtering respectively these effects can be used to give the idea of confinement or distance.

Cohen & Ludwig use thickening and self-animation to emphasise a sound source and muffling and distancing to de-emphasise background sources. These variable parameters are then used to create hierarchies of sounds in a similar way to those in hierarchical earcons. At the highest level thickening and self-animation are used to transform a sound; the next level uses just self-animation and the final level uses only the unprocessed sound. Thinning and muffling could also be used to add further depth to the hierarchy. Thus, five levels of a hierarchy could be represented without using any of the parameters suggested by Blattner *et al.* for earcons. These effects are strongly based around the theory of auditory pattern recognition (see Chapter 2). To help the listener group the sounds into sources the concepts of good continuation and coherence are used, i.e. changes within a sound source occur smoothly, continuously and in a coherent way, for example self-animation and thickening.

In Cohen & Ludwig [45] an interface was implemented using the ideas mentioned above. The system allowed the user to manipulate auditory sources via a DataGlove. Each sound source was given a virtual position with respect to the listener and the correct interaural time and intensity differences were calculated by the convolution engines. When a source was moved its intensity and time delay to the user were recalculated. Active audio filters, called *filtears*, were developed to indicate different states to the user. Cohen & Ludwig (p 325) suggest that “Our cues are like earcons … But unlike pure earcons, our cues are transforming, rather than transformed”, i.e. the filtears transform sounds which are being played, whereas with earcons the sounds are transformed statically and presented later. They indicate that filtears should be ‘just noticeable’; they should be transparent unless the user is actively seeking them so that they do not interfere with the sounds being presented. They implemented three different filtears:

- ❖ *An audio cursor or spotlight:* This was used to focus auditory attention, which is omni-directional, onto a particular source in the same way visual attention can be focused. The source could be highlighted using self-animation or pitch shifting. Cohen & Ludwig suggest that this could be used to indicate the selection of a sound source or pointing to it with the DataGlove.

- ❖ *Muffle:* This was used to indicate the grabbing of a source - a hand closed around a source muffles it. The DataGlove allowed the user to grab objects.

- ❖ *Accent:* This was used to emphasise a source so that a user would attend to it. This is different to a spotlight as this only highlights a source that it passes over whereas an accent would occur on the source all the time. As Cohen & Ludwig (p 327) indicate “Unlike spotlights or muffles, accents persist beyond pointing or grabbing.”

The filtears were used with the DataGlove to allow the user to select and grab objects and drag them to different places, with the correct localisational cues presented. Objects could be highlighted so that the user noticed them and could interact with them. Cohen & Ludwig (p 334) suggest that their interface has advantages because it “exploits our innate localisational abilities, our perception of spatial attributes and our intuitive notions of how to select and manipulate objects distributed in space". The system is strongly based on psychoacoustic principles. It uses intensity and time delays to provide full, three-dimensional localisation cues which make the user identification of sound sources more accurate than with just stereo information. The drawback of the system is that it involves the use of very expensive hardware and therefore would be difficult to add into the interface of a personal computer. However, some of the ideas on the active filtering of sounds may be applicable to more low cost systems, because simple filters and effects are present on most electronic music equipment. These filtering techniques could perhaps be applied to earcons to create more variable parameters and make them more easily distinguishable from one another.

Filtears could be used to generate a higher level of earcon parameters. The basic earcon parameters, discussed previously, describe a set of *static* earcons to represent items in an interface such as files, folders or menu items. These sounds are fixed so that a wordprocessor file will always have the same earcon. A set of *dynamic* earcon attributes can then be applied, based upon the ideas of Cohen, to provide a meta level of organisation. For example, the wordprocessor file might be in a background window so that it makes a muffled sound if the mouse is moved over it. This would be dynamic as, if the window was brought to the front and the icon selected, the sound produced would change. It might then be accented to indicate that it was the focus of attention. The static parameters would not change but the dynamic ones would depending on the state of the interface. They could be used to group earcons in other ways, such as foreground and background sounds. This is an area that needs further investigation.

3.12 SOME EXISTING GUIDELINES FOR THE USE OF SOUND

There are two existing sets of guidelines for the use of sound and both of these are for warnings in aircraft cockpits. Even though desktop computer interfaces are very different environments to aircraft cockpits, many of the guidelines are transferable. The first set of guidelines discussed, by Patterson and colleagues, are for general auditory feedback design in cockpits and the second, by Edworthy and colleagues, are for predicting the urgency of auditory warnings.

Patterson [128, 129] investigated some of the problems with auditory warnings in aircraft cockpits. Many of the warnings were added in a 'better safe than sorry' manner which lead to them being so loud that the pilot's first response was to try and turn the alarm off rather than deal with the problem it was indicating. Patterson also says that the sounds were not added in a coherent way, there was no overall method behind the warnings which lead them to conflict with each other. To overcome these problems Patterson produced a set of guidelines covering all aspects of warning design. For example, the warnings he suggests use much lower intensities and slower onsets/offsets to avoid startling the pilot. The main points addressed by his guidelines are:

- ❖ *Overall level:* The lowest intensity level of a warning sound should be 15dB above the threshold imposed by background noise. The upper limit is 25dB above the threshold.

- ❖ *Temporal characteristics:* Component pulses of a warning sound should have onsets and offsets 20-30 ms in duration. These avoid a startle response in the listener. Pulses should be 100-150 ms in duration with a 150 ms inter-pulse gap for urgent sounds and a 300 ms gap for non-urgent sounds. Distinctive rhythms of five or more pulses should be used.

- ❖ *Spectral characteristics:* The fundamental frequency of a warning should be within the range 150-1000 Hz. There should be four or more component harmonics to help avoid masking. The overall spectral range of warnings should be 500-5000 Hz.

- ❖ *Ergonomics:* Manual volume control should be avoided and automatic control restricted to a range of 10-15 dB variation. There should be no more than six immediate action warnings.

- ❖ *Voice warnings:* These should be brief and use a key-word format. They should not be repeated in a background version of the warning. Voice warnings used as immediate awareness warnings should use a full-phrase format and be repeated after a short pause.

Patterson has used these guidelines to suggest changes in the sounds used in intensive-care wards of hospitals [131] and other work areas that use sound [130]. Patterson showed that warnings designed around his guidelines were less likely to be confused than those currently used in aircraft. His research provides a base from which to work and these guidelines are used in creating the sound stimuli in the following chapters.

Edworthy, Loxley, Geelhoed & Dennis and Edworthy, Loxley & Dennis [64, 65] have experimentally shown that the level of perceived urgency of sounds can be predictably controlled. They manipulated spectral and temporal components of sound to change the urgency [64]. The results of their work showed that higher fundamental frequencies created greater perceived urgency. Slow onset and offset times produced lower perceived urgency. Irregular harmonics were perceived as more urgent than regular ones. Speed of presentation of stimuli was also important; rapidly presented stimuli were perceived as more urgent that slowly presented ones. Stimuli which speeded up when presented were more urgent than those that slowed down or did not change.

Regular rhythms were more urgent than syncopated ones. Stimuli with a greater the number of repeated units of sound were perceived as more urgent than stimuli with fewer repeating groups. Stimuli that covered a large pitch range were more urgent than those with a small range. Atonal sounds were more urgent than more musical ones. For a detailed description of each of the parameters see [64].

Edworthy *et al.* conducted one other experiment to test the predictability of each of the parameters described above. They created twelve stimuli that used different combinations of the parameters and predicted their order of perceived urgency. The results showed that only one stimulus was not at the predicted urgency level. Therefore, the urgency of the auditory warnings was highly manipulable and predictable. They also showed that there was strong agreement between subjects on the urgency of the sounds used. As Edworthy *et al.* ([65], p 80) say: “The results may be applied systematically and reliably to the design of future auditory warning systems”. These guidelines allow the creation of sounds that will be perceived as very urgent for high priority situations and not urgent for low priority situations. This is very important for sound stimuli in auditory interfaces. Errors could be indicated with very urgent sounds, warnings with less urgent ones and informational messages with non-urgent sounds.

3.13 CONCLUSIONS

This chapter has provided a background for the use of non-speech audio at the interface. It has shown that there were problems with synthetic speech which meant non-speech sounds were advantageous. A strong psychological basis for using a combination of sound and graphics for presenting information at the interface was described. The chapter dealt with some of the major criticisms of sound and showed that many of these do not apply to carefully constructed non-speech sounds. Several systems for presenting information in sound were discussed in detail, especially earcons and auditory icons. The review described several examples of systems using auditory icons but very few systems using earcons. This shortage of work on earcons provided the motivation for the rest of the work in this thesis, where earcons are investigated in greater detail. The lack of specific guidelines for the creation of earcons was shown to be a problem as systems have been produced using earcons that were not effective. Blattner *et al.* suggested some general guidelines but these have not been tested as they never implemented a system of earcons. More specific guidelines that have been experimentally verified are needed if designers are to use earcons to create auditory interfaces.

Most of the systems described did not provide any method for finding where in the interface sounds would be useful, they just suggested methods of presenting information in sound. Gaver suggested that auditory icons should be used in places suggested by the natural environment. This was shown to be a problem as there were many situations in the interface where there were natural equivalents. Providing a method to find where sound might be useful is another area addressed later in this thesis. This chapter has shown that there is a growing body of work in this new field. However, there are still many aspects that need further investigation.

CHAPTER 4: AN INVESTIGATION INTO THE EFFECTIVENESS OF EARCONS

4.1 INTRODUCTION

In this chapter two experiments to assess the effectiveness of earcons for presenting information in sound are discussed. These attempt to answer the question: What sounds should be used at the interface? There are potentially many different types of sounds that could be used. The previous chapter described earcons, auditory icons and synthetic speech. The latter was ruled out because of problems such as slow presentation rates and demands on short-term memory. Auditory icons have problems because often there are no real world sounds to match an abstract operation or action in an interface. This chapter investigates earcons which are potentially very powerful but have never before been tested.

First of all, the motivation for investigating earcons is described and then the two experiments to assess their viability. The first experiment is an exploratory study to test the effectiveness of the rules for earcon creation put forward by Blattner *et al.* [25] and the second builds on these rules to overcome some of the problems brought to light. A set of guidelines are put forward, based on the experiments, to help designers create effective earcons. These form half of the structured method for integrating sound into human-computer interfaces.

4.2 MOTIVATION FOR INVESTIGATING EARCONS

In Chapter 3 two different methods for presenting information in sound were described: *Earcons* (Blattner, Sumikawa & Greenberg [25]) and *Auditory Icons* (Gaver [73]). Although no formal analysis has been undertaken, auditory icons have been shown to be effective at communicating information in several different applications and environments. Earcons, on the other hand, have had much less work done on them. Blattner *et al.* did not create any earcons to test their original ideas. As Sumikawa ([164], p 97) says: “The next step in the continuation of this research is the development of an actual implementation of a computer/user interface employing earcons.” There were no examples of interfaces using earcons in any published work when the research described here was begun. Even now, at the end of this project, there is only a small amount [14, 26] (apart from that described here). Earcons appeared to be a useful way to present structured information so it was decided that an investigation into their effectiveness was needed.

In the previous chapter some of the problems faced by earcon designers were described. Barfield *et al.* [14] designed a menu system using earcons to aid navigation but the sounds failed to make any difference. The earcons they used were very simple, just decreasing in pitch as a subject moved down a hierarchy. These sounds did not fully exploit all the advantages offered by earcons (for example they did not use rhythm or different timbres) and they did not improve performance. If better earcons were used then a performance increase may have been found. If earcons are created without care then they may be ineffective. Portigal [136] also used abstract sounds, similar to earcons, in his work on displaying document structure in sound. His research failed to show any advantages from using sound. These two sets of results suggest that research is needed to discover if earcons are an effective means of communicating in sound or not. It may be that they are too complex for listeners to understand. Work is needed to put forward a detailed set of guidelines to help future designers of earcons make sounds that will communicate their messages effectively.

This chapter describes two experiments to investigate earcons. The first experiment tried to answer three main questions. The first was: Are earcons a good method of communicating complex information in sound? This was investigated by looking at the overall recognition rates of earcons. The second question was: Are musical timbres more effective than simple tones? Blattner *et al*. [25] suggest the use of simple timbres such as sine or square waves but psychoacoustics suggests that complex musical instrument timbres may be more effective. This was investigated by comparing earcons with musical timbres against earcons with simple tone timbres. The final question was: Is rhythm important in the recognition of earcons? Standard system beeps do not use rhythm information but Deutsch [49] says this is one of the most important characteristics of a sound. Blattner *et al.* [25] describe this as the most prominent characteristic of a motive. This was investigated by using a control group with no rhythm information. The second experiment used the results of the first to create new earcons to overcome some of the difficulties that came to light. Guidelines are then put forward for use when creating earcons. The research described in this chapter has appeared in two papers [32, 33].

4.3 EXPERIMENT 1

In order to assess the effectiveness of earcons an experiment was needed. This would test to see if subjects could recall and recognise earcons. The experiment would not attempt to suggest where sounds should be used at the interface but just if earcons were an effective means of presenting complex information. There were four phases to the experiment. In phase I subjects heard earcons for icons, in phase II they heard earcons for menus, phase III was a re-test of phase I and phase IV was a test of combined earcons made up from phases I and II. There were three groups of subjects: The musical group, the simple group and the control group.

4.3.1 Subjects

Thirty-six subjects, three groups of twelve, were used from the University of York. Seventeen of the subjects were musically trained (they could play an instrument and read music). They were randomly allocated to one of three groups so that there was an even mix of musicians and non-musicians (the simple tone group had only five musicians).

4.3.2 Sounds used

An experiment was designed to find out if structured sounds such as earcons were better than unstructured sounds for communicating information. Simple tones were compared to complex musical timbres. Rhythm was also tested as a way of differentiating earcons. According to Deutsch [49] rhythm is one of the most powerful methods for differentiating sound sources. The experiment also attempted to find out how well subjects could identify earcons individually and when played together in sequence.

Figure 4.1 and Figure 4.2 give the rhythm and pitch structures used in phases I and II of the experiment. The rhythms for open/close and delete/create were adapted from Blattner *et al.* The sounds were based on the general guidelines proposed by Sumikawa, Blattner, Joy & Greenberg [163]. These suggested a maximum number of four notes. They do not suggest any tempo so 180 beats per minute was used making the longest earcon 1.32 seconds.

q = 0.33 seconds giving a tempo of (60/0.33) = 180 beats per minute (bpm).

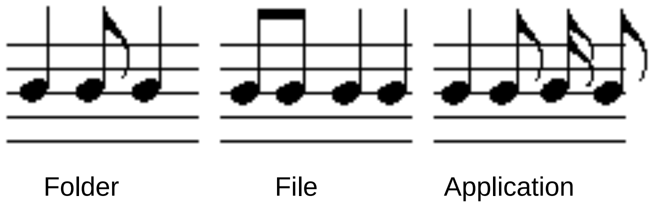


Figure 4.1: *Rhythms used in phase I of Experiment 1.*

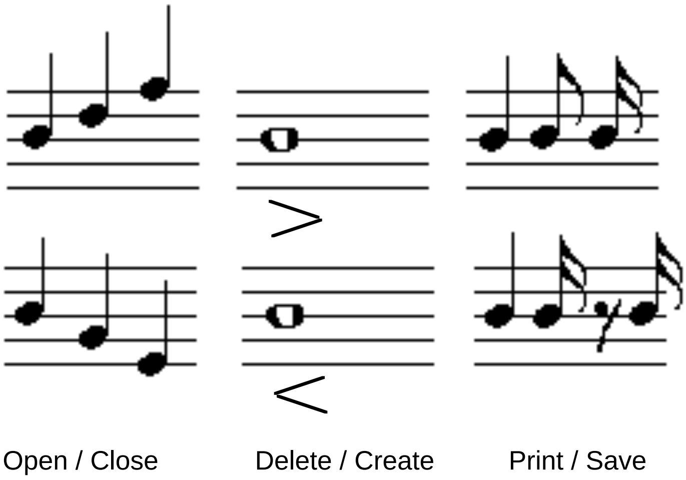


Figure 4.2*: Rhythms used in phase II of Experiment 1.*

The sounds in phase I were based around C_3 (middle C). There were three types of earcons: One for files, one for folders and one for applications (see the section below for more on phase I). The earcon for an application was at C_3, the folder sound at C_2 and the file 1 sound at C_4. This pitch spread was used to give greater differentiation of the earcons. It also allowed rhythm information to be taken out of the control group but for the sounds to still be different. In phase II the sounds were based around C_2.

Three sets of sounds were created (Table 4.1 gives more information):

❖ *Musical Sounds*: The first set were synthesised complex musical timbres: Piano, brass, marimba and pan pipes. These were produced by a Roland D110 multi-timbral sound synthesiser. This set had rhythm information as shown in the figure above.

❖ *Simple Sounds*: The second set were simple timbres: Sine wave, square wave, sawtooth and a ‘complex’ wave (this was composed of a fundamental plus the first three harmonics. Each harmonic had one third of the intensity of the previous one). These sounds were created by SoundEdit on an Apple Macintosh. This set also had rhythm information as shown above.

- *Control Sounds*: The third set had no rhythm information; these were just one-second bursts of sound similar to normal system beeps. In phase I, this set had timbres made up from the musical group and were differentiated by the pitch structure described above. In phase II, different timbres were used to differentiate the different menus. These were different to the timbres used in phase I to avoid confusion. If different timbres were not used then there would be nothing to differentiate the phase I sounds from the phase II ones. This was possible in the other two groups because of the different rhythms used. See Table 4.1 for the timbres used.

The simple tones used were based around the guidelines suggested by Blattner *et al.* [25] (see Chapter 3). These gave only general advice. When choosing musical timbres, trial and error was used as there were no precise rules on which a choice could be based. As described in Chapter 2, there is no complete set of descriptors for timbre. This problem meant that there was no way to say how close one timbre was to another and they could not be varied in systematic ways. Therefore, when choosing timbres, one was tried and it was subjectively assessed by the experimenter to see how 'different' it sounded to the others chosen. As described below, the experiment was prototyped on several subjects before the main test was conducted and after their comments other timbres were chosen to aid recognition.

	Musical Group	Simple Group	Control Group
Write	Piano	Square	Piano
Paint	Brass	Sine	Brass
Draw	Marimba	Sawtooth	Marimba
HyperCard	Pan Pipes	Complex	Pan Pipes
Menu 1	Piano	Sine	Flute
Menu 2	Pan Pipes	Complex	Electric Organ
Menu 3	Marimba	Square	Cymbal

Table 4.1: *Timbres used in the different groups in Experiment 1.*

HyperCard on an Apple Macintosh computer was used to play the sounds. In the musical group the sounds were played on a Roland D110 synthesiser, as described above. The synthesiser was controlled via MIDI from the Macintosh. The sounds were all played through a Yamaha DMP 11 digital mixer and presented using external loudspeakers.

4.3.3 Experimental design and procedure

The format of the experiment is shown in Table 4.2. An independent-subjects design was employed. As mentioned, three groups of twelve subjects were used. Each of the three groups heard different sound stimuli. The musical group heard the musical sounds described in the previous section. The simple group heard the simple sounds and the control group heard the control sounds. There were four phases to the experiment. In the first phase subjects were trained and tested on sounds for icons (objects). In the second they were trained and tested on sounds for menus (actions). In the third phase they were re-tested on the icon sounds from phase I. In the final phase subjects were required to listen to two earcons played in sequence and give information about both sounds heard. Instructions were read from a prepared script. A prototype version of the experiment was tested on four subjects to make sure the stimuli were correct and there were no errors in procedure. Any problems found were corrected before the main experiment was run.

Subjects	Sounds	Phase I	Phase II	Phase III	Phase IV
Musical Group	Musical Sounds	Icon Sounds Train & Test	Menu Sounds Train & Test	Icon Sounds Re-Test	Combined Menu & Icon Sounds Test
Simple Group	Simple Sounds				
Control Group	Control Sounds				

Table 4.2: *Format of Experiment 1.*

In the training for the experiment subjects were not told the names of the timbres used; they had to make up their own labels. Corcoran, Carpenter, Webster & Woodhead [47] suggest that this would produce a low rate of recognition. A better method, they propose, would be to use a set of expert-generated names. In their research these produced the best result. However, the experiment here was designed to be a ‘worst case’ evaluation of earcons so it was decided that users should generate their own. If earcons proved to be effective under these conditions, then in a situation where more structured training was given results would be higher.

4.3.4 Phase I

Training: The subjects were presented with the screen shown in Figure 4.3. The icons were described to each subject. The relationships between types were described. For example, the relationship between programs was indicated by having icons with hands in the graphic. The relationships between families were described and all the members of each family pointed out. The subjects were then asked to learn the names of all the icons. When they thought they had done this they were required to write them down. If they were not correct they were allowed more time to learn them. This meant that, at the end of the training, the subjects knew the names of all the icons present.

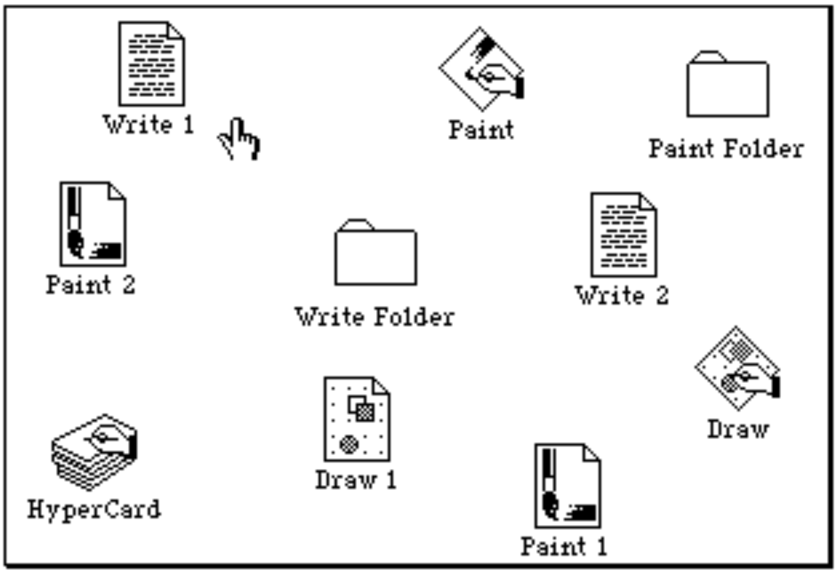


Figure 4.3: *The phase I icon screen.*

Each of the objects on the display had a sound attached to it. The sounds were structured as follows: Each *family* of related items shared the same timbre. For example, the paint program, the paint folder and paint files all had the same instrument. Items of the same *type* shared the same rhythm. For example, all the programs had the same rhythm. Items in the *same* family and of the *same* type were differentiated by pitch. For example, the first Write file was at pitch C_4 and the second Write file was at G_5. In the control group no rhythm information was given so types were also differentiated by pitch. Figure 4.4 shows the hierarchy of earcons used in phase I. The different families were differentiated by timbre. Within the different families, the different types of icons were differentiated by rhythm. Different files were differentiated by pitch alone. In Figure 4.4 Level 2 of the graph has been expanded to avoid cluttering the diagram with many crossed lines. As described above, the rhythms used by each *type* of item were the same so, for example, the program rhythm used by the Paint, Draw and HyperCard families was the same. All of the information available graphically in the icons was available through sound in the earcons.

The mapping between the sounds and the icons was described to the subjects. However, the subjects were not told the precise rhythms and timbres used for each item. For example, subjects were told that all items in the Paint family had the same timbre but not what that timbre was. The earcons were then played one-at-a-time in random order by the experimenter clicking on each of the icons in turn. The whole set of earcons was played three times. The random order was the same across the three presentations to each subject and was the same to all of the subjects.

The random nature of the training made the sounds harder to learn. If, for example, paint file, paint folder and paint application had been played one after another then the relationships would have been easier to identify and it would have been easier for the subjects to form a model of the hierarchy used [47]. Corcoran *et al.* indicate that "…sounds that the trainee is likely to find difficult to distinguish should be presented alternately, not separated in time by other intervening sounds". However, doing it in the way described here gave a 'worst-case' learning situation. If the earcons could be learned under these conditions then they would be a robust method of communication.

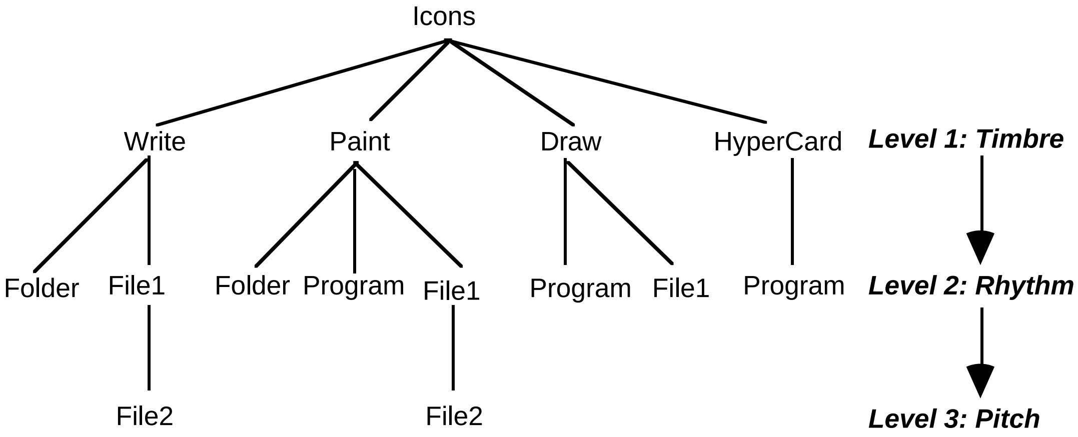


Figure 4.4*: The hierarchy of earcons used in phase I of Experiment 1.*

Testing: When testing the subjects the screen was cleared and a selection of the earcons were played back in random order. The subject had to supply what information they could about type, family and if it was a file then the number of the file. The testing was again the 'worst-case' situation. Subjects were given no visual cues when the sounds were played back. In an interface where sound and graphics were integrated, when a sound was heard there would be visual feedback to aid recall. For example, if sound was added when scrolling with a scrollbar then visual information would be present to aid recall of the sound. This experiment tested the 'worst-case' in training and testing to see what could be achieved. In this and all the phases the subject was allowed to hear each sound a maximum of three times. Ten questions were asked. When scoring, a mark was given for each correct piece of information supplied. In the testing of this and the other phases subjects were not told the accuracy of their responses.

4.3.5 Phase II

In this phase earcons for menus were tested. Each *menu* had its own timbre and the *items* on each menu were differentiated by rhythm. Pitch and intensity were also used to create a rich mixture of stimuli. The screen shown to the users to learn the earcons is given in Figure 4.5. The hierarchy of earcons used in this phase is shown in Figure 4.6. The training was similar to phase I. The subjects were tested in the same way as before but this time had to supply information about menu and item. Ten questions were asked (one earcon being repeated).

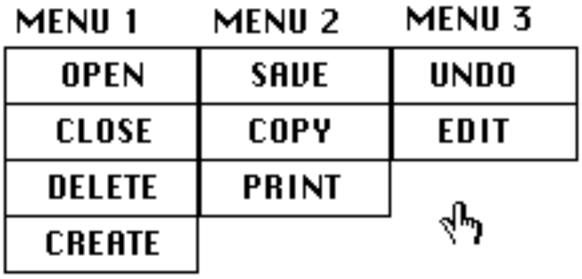


Figure 4.5: *The phase II menu screen.*

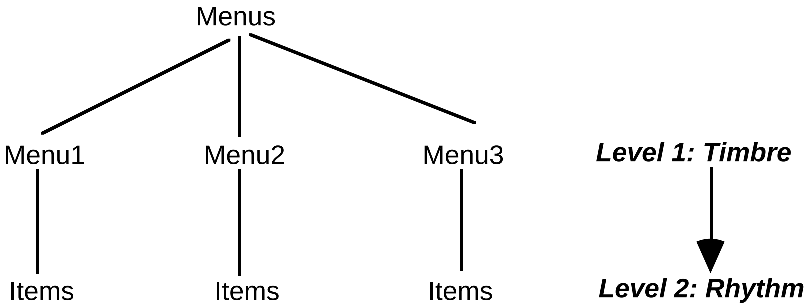


Figure 4.6*: The hierarchy of earcons used in phase II of Experiment 1.*

4.3.6 Phase III

This was a re-test of phase I but no further training time was given and the earcons were presented in a different order. Ten questions were asked. This was to test if the subjects could remember the original set of earcons after a period time and having learned another similar set of stimuli.

4.3.7 Phase IV

This was a combination of phases I and II. Again, no chance was given for the subjects to re-learn the earcons. The subjects were played two earcons, one followed by another, and asked to give what information they could about each sound they heard. The sounds heard were from the previous phases and could be played in any order (i.e. it could be menu then icon, icon then menu, menu then menu or icon then icon). This gave a maximum of (19*18) 342 possible combinations of earcons (as no two same earcons were played together). This would test to see what happened to the recognition of the earcons when played in sequence. When testing, twelve questions were asked. The thirteenth stimulus played had three earcons in it. A mark was given for any correct piece of information supplied.

4.3.8 Experimental hypotheses

The main aim of the experiment was to discover if earcons could convey complex messages in sound. If they could not then there would be no point in using them; alternative methods would have to be found. The main hypothesis was that earcons would be effective at communicating complex information. The overall recognition rates of earcons would be high and higher than the less structured control sounds. The structured nature of earcons would make them easier to remember and discriminate than unstructured sounds such as standard system beeps. In terms of results the musical and simple groups should have high overall recognition rates and they should be significantly higher than the control group (which used less structured sounds).

Psychoacoustics (see Chapter 2) suggests that musical timbres should be more easily recognised than the simple tones proposed by Blattner *et al.* [25]. This is due to their greater spectral and temporal complexity making them more discriminable than simple tones. Therefore recognition of timbre information would be better in the musical group than in the simple group in all phases (the simple group used simple tones, the musical group complex musical ones).

The rhythms in the simple and musical groups would increase the recognition of type. The score in these two groups would be significantly higher than the score in the control group (which used no rhythm information). Pitch alone would make it hard for subjects to identify earcons. Making absolute pitch judgements is difficult except for the few people have ‘perfect pitch’ (see Chapter 2 for more on this). Therefore file sounds in phase I would be harder to tell apart in the simple and musical groups and recognition would be lower than for the timbres and rhythms.

In order to find out if memory for earcons was strong phase III re-tested the phase I earcons after the subjects had learned the phase II earcons (a short period of time had also passed). If memory for earcons was strong then recognition rates between phase I and III would be unchanged.

One of the advantages suggested by Blattner *et al.* [25] was that earcons could be combined into compound messages. Compound earcons of two components would be as recognisable as single earcons due to their structure making them easier to remember. Therefore, the overall scores in phase IV would not be significantly different to those of the other phases. There would also be no difference between the recognition rates of the two earcon compounds and three earcon compounds. Compound earcons of three components would be as easy to recognise for the same reasons as above.

One other important issue was to find out if earcons could only be used by skilled musicians and not non-musicians. If this was the case then the practical uses of earcons would be limited. The hypothesis was that there would be no difference between these two types of subjects because recognition would be based on basic human perceptual properties and not musical skill.

4.4 RESULTS AND DISCUSSION OF EXPERIMENT 1

When marking the answers a mark was given for every correct piece of information in a response. For example, if a subject replied with ‘Paint Program’ when the answer was ‘Paint Folder’ then one mark was given but if the correct answer was given then he/she got two marks. For the detailed data analysis these individual component scores were used. These raw data were also used to construct a new set of scores for overall earcon recognition. The component scores were simply added together to produce a total score. This new set of data was used for the overall analysis described below and is different to that used for the component analysis. The raw results data are shown in Appendix A Table 1.

An overall analysis was performed first. The main part of the analysis would focus on the component data but an overall analysis would show any overall effects of the different types of sounds. Figure 4.7 and Figure 4.8 suggest that musical earcons produced higher percentage correct scores than simple earcons which in turn were better than the control sounds. However, these differences were not significant. The graphs show the percentage of correctly identified parts of the earcons for each of the groups. The experiment was a multifactorial (3x4) mixed design. A between-groups across-phase analysis of variance [85] (ANOVA) was carried out on the overall data (the data in Figure 4.8). The ANOVA test allows the evaluation of differences in means between several groups [145]. This test allows the investigation of the effects of phase and group in one analysis. It also allows the investigation of any interactions between phase and group that were not shown in the means.

The ANOVA showed no main effect for group (F(2,33)=1.17, p=0.3231). There was, however, a main effect for phase (F(3,33)=23.48, p=0.0001) but no interaction between group and phase (F(3,33)=1.81, p=0.1055). The hypotheses proposed that the musical group should have performed best overall as it had musical timbres and rhythm information, but this was not the case. The overall recognition rates for the earcons were not significantly different between the groups. The percent correct scores were: 58% (musical group), 53.85% (simple group) and 48% (control group). The earcons did not reach a high overall level of recognition and they were not significantly better than the less structured control group sounds. These results indicate that earcons were not effective at communicating complex information in sound and this hypothesis was rejected. The overall results, however, may have been masking important differences within the gross data. Therefore, a more detailed analysis of each phase was needed to find out if there were any differences hidden within the overall data.

The ANOVA showed that there were differences in the phase scores but not where they occurred. In order to find out what caused the main effect for phase post-ANOVA

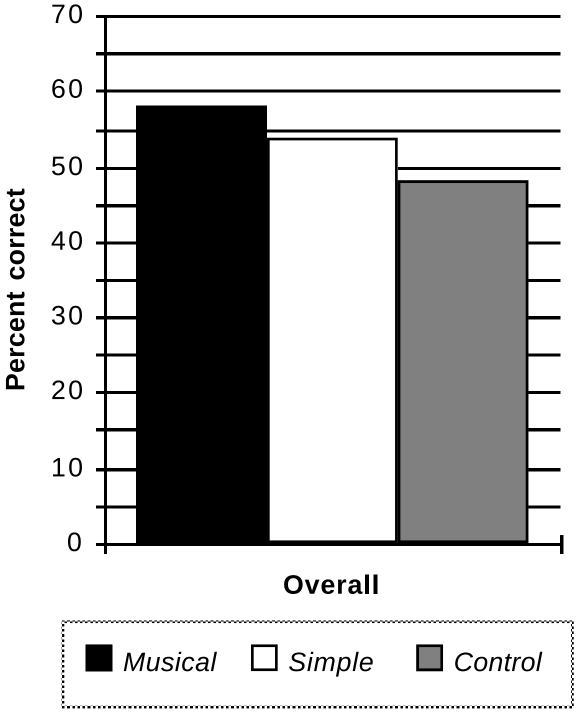


Figure 4.7: *Overall scores in Experiment 1.*

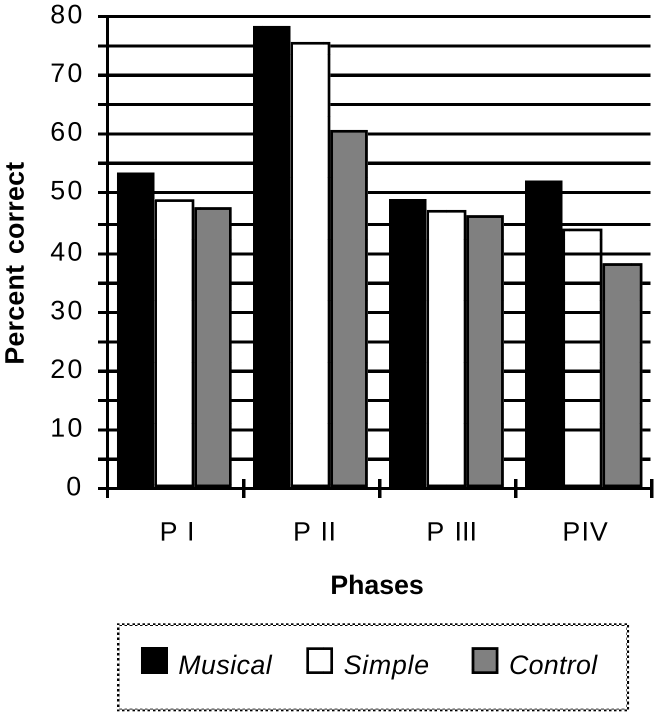


Figure 4.8: *Breakdown of overall scores per phase for Experiment 1.*

Tukey HSD tests [145] were performed on the phase data for each group. These tests allow us to compare the different phases whilst retaining a high level of confidence that we are accepting a difference which is real. This would not be the case if other tests (such as the T-test [144]) were performed in this situation. An alternative to the Tukey HSD test is the Scheffé F-test. This test is also used later in the chapter and works in the same way.

The results of the Tukey tests are shown in Table 4.3. Only the significant test results are shown; any combinations that are not shown were not significant. They show that in each group phase II was always significantly better recognised than phase IV. In fact, in the musical and simple groups phase II was significantly better recognised than all of the other phases. There were no significant differences between any of the other phases. In the control group phase I was also significantly better recognised than phase IV. This indicated that the compound earcons were the hardest to recognise for this group. The hypothesis that compound earcons would be as easy to recognise as individual earcons cannot be accepted or rejected until further analysis has taken place. These results clearly showed that the phase II stimuli were much more effective than the other phases and further detailed investigation was needed to find out why this was.

Phases	II vs. I	II vs. III	II vs. IV	I vs. IV
Musical Group	Q(33)=6.581, p=0.01	Q(33)=6.04, p=0.01	Q(33)=5.245, p=0.01	
Simple Group	Q(33)=7.62, p=0.01	Q(33)=6.86, p=0.01	Q(33)=6.75, p=0.01	
Control Group			Q(33)=5.22, p=0.01	Q(33)=4.317, p=0.05

***Table 4.3**: Tukey HSD tests conducted on the percentage scores of the overall phase data for each group.*

4.4.1 Phase I

A detailed analysis of the component data for each phase was undertaken. This would allow the comparison of the individual components (family, type and file) across the different groups to investigate the effects of the sounds. Creating the overall scores from the individual components may have masked some differences within the data. Therefore a more detailed analysis was needed to avoid this.

In phase I the breakdown of scores can be seen in Figure 4.9. A between-groups ANOVA was carried out on the family scores (family was differentiated by timbre) and showed a strongly significant effect (F(2,33)=9.788, p=0.0005). Post-ANOVA Scheffé F-tests [85] showed that the family score in the musical group was significantly better than the simple group (F(2,33)=6.613, p=0.05) and that the control group was better than the simple group (F(2,33)=8, p=0.05). In this phase, both the musical and control

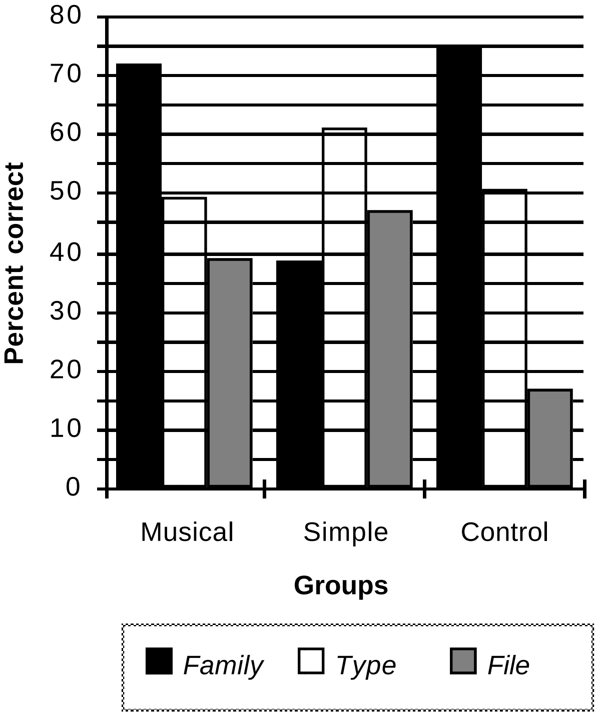


Figure 4.9: *Breakdown of scores for phase I of Experiment 1.*

groups used musical timbres (see Table 4.1). These two results indicate that musical instrument timbres were more easily recognised than the simple tones proposed by Blattner *et al.* [25]. This confirms the hypothesis that musical timbres would be the most effective. The poor performance on simple tones confirms work by the European Telecommunications Standards Institute (ETSI) [66, 67]. The ETSI standards for audible tones in telephone systems suggest that if simple tones are used then recognition rates will be low and listeners will only be able to recognise between four and six tones. This once again mediates against the use of simple tones at the human-computer interface.

There were no significant differences between the groups in terms of type (differentiated by rhythm plus pitch in the musical and simple groups and just pitch in the control group). A between-groups ANOVA showed no effect (F(2,33)=1.21, p=0.312). According to the hypotheses, the control group should have performed the worst as it had no rhythm information. However, the results show that the simple and musical groups performed no better. Therefore, the rhythms used did not give any better performance over pitch alone: The chosen rhythms were ineffective. The hypothesis that rhythm improves recognition rates was rejected.

One final analysis of the data from phase I was undertaken. The type, family and file components were compared to see if there were any differences in rates of recognition between them. The hypotheses predicted that pitch on its own would not be an effective method of providing information. Testing file scores against family and type scores would show whether this was the case. For this analysis percentage data was used. A within-groups one-factor ANOVA was performed on the file, family and type scores for each group separately. The results showed a significant effect for the musical and control groups (musical group: F(2,33)=4.544, p=0.01, control group: F(2,33)=22.54, p=0.0000006) but not for the simple group (F(2,33)=2.75, p=0.07). To find out where the effects occurred in the musical and control groups Tukey HSD tests were performed between the different components. In the musical group family scores were significantly better than file scores (Q(33)=4.17, p=0.05). In the control group family was significantly better than file (Q(33)=9.46, p=0.01) as was type (Q(33)=5.54, p=0.01). These results again show that pitch alone is not a good method for differentiating earcons. These results confirm the hypothesis that pitch used in the way described here is not a good method of communicating information.

There was no difference between the type scores and the file scores in the musical and simple groups. This was expected because the type analysis described above showed that the rhythms plus pitches used for type were no more effective than pitch alone used in the control group. It is unclear why the subjects recalled type significantly better than file in the control group as both were differentiated by pitch. It may have been that the subjects put most of their effort into recognising the types and so could not recognise the files. This was supported by the low file scores in this group.

These results confirm those of Edwards [62] on Soundtrack, described in detail in Chapter 3. He used different pitches to indicate various areas of the screen. Users moved around and heard a different pitch to indicate where they were on the screen. His hypothesis was subjects would be able to tell where they were from the pitch of the tone they heard. His results showed that subjects could not use this pitch information effectively. Thus, pitch on its own is not a useful method for differentiating sounds. Absolute judgements of pitch are difficult [39] unless listeners have *perfect pitch*. In order for the subjects to identify which file sound was played they had to make an absolute pitch judgement. Subjects could not compare the two sounds and make a relative judgement, which would have been easier. One other reason for there being low scores for pitch could be that the 'file 1' earcons were at C_4 (130Hz) and 'file 2' earcons at G_5 (98Hz). 98Hz was below the minimum pitch suggested by Patterson [128] in his guidelines for auditory warning design (see Chapter 3).

4.4.2 Phase II

As described above (see Table 4.3), this phase was significantly better than all of the others. The factors that caused this were investigated further. Figure 4.10 shows the results for this phase. A between-groups ANOVA was carried out on the item scores (item was differentiated by rhythm) and it showed an effect for group (F(2,33)=9.65,

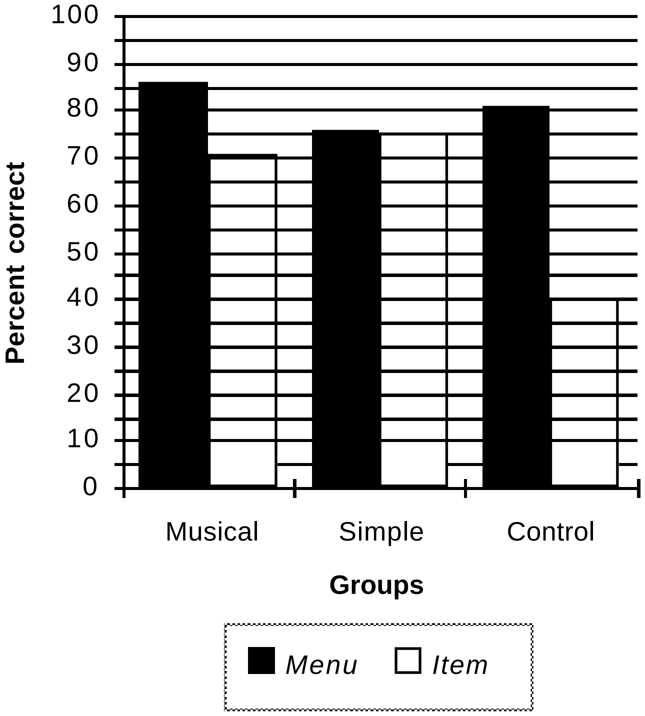


***Figure 4.10**: Breakdown of scores for phase II of Experiment 1.*

p=0.0005). Scheffé F-tests showed both the musical and simple groups were significantly better than the control group (musical vs. control F(2,33)=6.278, p=0.05, simple vs. control F(2,33)=8.089, p=0.05). There was no difference between the musical and simple groups as they both used the same rhythms (F(2,33)=0.11, p=0.05). This shows that if rhythms are chosen correctly then they can be very important in aiding recognition. This confirms research by Deutsch [49] (see Chapter 2). This proves the hypothesis that rhythm can significantly improve recognition. This improvement in the recognition of rhythm was the main reason for the significantly better overall score in phase II than in phase I. For example, in the musical group recognition of rhythm only reached 49% in phase I but in phase II it reached 71%.

These results again show that pitch alone is not as good a discriminator as rhythm. The control group only reached 40% recognition in phase II whereas the musical and simple groups reached 71% and 75% respectively. This confirms the hypothesis that pitch is not a good parameter to use to make differentiable earcons.

A between-groups ANOVA was carried out on the menu scores (differentiated by timbre), it showed no effect (F(2,33)=1.88, p=0.1691). This goes against the scores in the previous phase where there was a significant advantage for the musical timbres over the simple tones. The menu scores for the musical and control groups were high in this phase. The simple tones were raised to the level of the musical timbres rather than the musical timbre recognition falling. The recognition rates in phase II were at approximately the same level as in phase I. For example, the musical group in phase I reached 72% and in phase II 86%. It may have been that, in this phase, the earcons were generally easier, hence the overall higher scores, so that the subjects could better recognise the simple timbres. In this phase there was less information to remember: Only menu and item for three menus rather than four icon families with up to four components.

Phase II scored the highest recognition rates in any of the phases. The results were better than phase I because of significantly better recognised rhythms. The timbre scores were approximately equal. This suggests that if the rhythms in phase I were redesigned recognition rates could be increased to that of phase II.

4.4.3 Phase III

In this phase the earcons from phase I were tested again. A period of approximately 15 minutes had passed since the subjects learned the earcons. The scores in phase III were not significantly different to those in phase I (see Figure 4.8): There was no decrement in performance. The profile of the results was the same as phase I. This indicated that subjects managed to recall and recognise the earcons from phase I even after learning the sounds for phase II, which were very similar. This result confirmed the hypothesis that subjects would remember the earcons and showed that a subject’s memory for earcons was strong.

4.4.4 Phase IV

The results in Table 4.3 show that there were no significant differences between the overall scores in phases I, III and IV. Phase II was significantly better than phase IV. These results indicate that, for the compound earcons recognition rates were unchanged from when the phase I earcons were heard alone. Figure 4.11 shows the scores in phase IV where combinations of earcons were tested. A more detailed analysis was undertaken to discover any differences hidden in the overall data.

The three groups were tested to see if there were any differences in recognition between them on the individual components in phase IV. A between-groups ANOVA on each of the components showed the only significant effect to be on item score (item was differentiated by rhythm). Table 4.4 shows the ANOVA results. A Scheffé F-test showed that the item score in the musical group was significantly better than the control group ($F(2,33)=3.647$, $p<0.05$). Items were differentiated by rhythm in the musical group but the control group used pitch. This again confirmed the hypothesis that pitch alone was not as good a method for differentiating earcons as rhythm. This is similar to the phase II result where item was significantly better recognised in the simple and musical groups than in the control group.

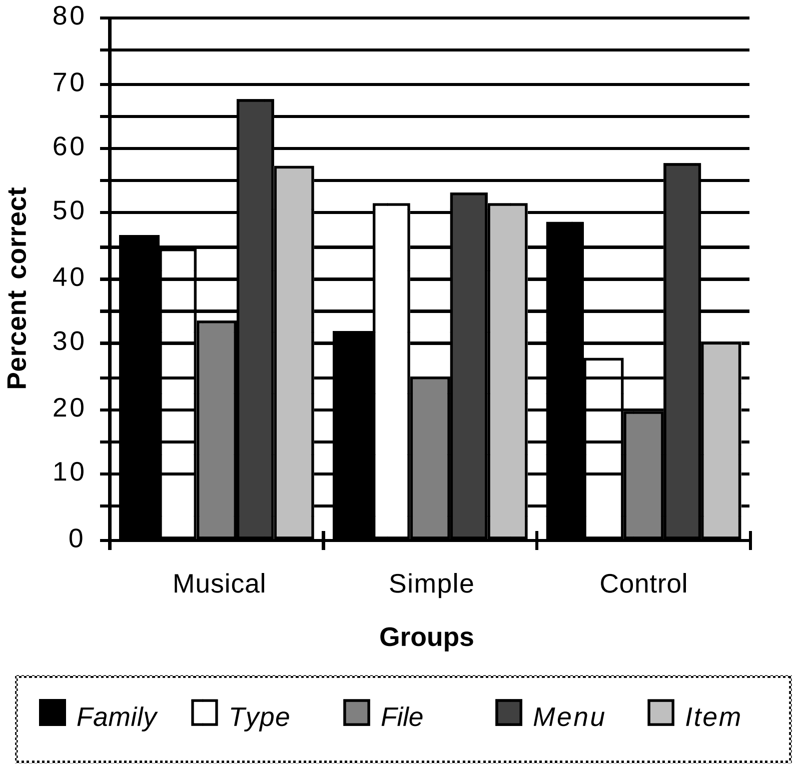


Figure 4.11: *Breakdown of scores for phase IV of Experiment 1.*

Family	Type	File	Menu	Item
F(2,33)=2.99, p=0.0642	F(2,33)=1.5, p=0.238	F(2,33)=0.6, p=0.556	F(2,33)=1.41, p=0.2582	F(2,33)=4.04, p=0.0269 *

Table 4.4*: Between-groups ANOVA results for each of the components in phase IV. Significant results are marked with an asterisk.*

In order to find out if recognition rates changed when earcons were combined, the individual components of phase IV were compared to their equivalents in the previous phases. For example, phase I type, family and file were compared to the phase IV scores for each of the groups separately. The overall analysis showed that there were no differences between phases I and IV but that phase II was significantly better than IV. For this within-group, across-phase analysis paired T-tests were used. One-factor analyses of variance tests could have been used as an alternative but, as Gravetter & Wallnau suggest [85], there is no difference between the two types of test in this situation. The T-test is a simple test that allows the evaluation of the difference between two means [144]. The results are shown in Table 4.5.

Phases / Groups	Phase I type vs. Phase IV type	Phase I family vs. Phase IV family	Phase I file vs. Phase IV file	Phase II menu vs. Phase IV menu	Phase II item vs. Phase IV item
Musical	T(11)=0.797, p=0.441	T(11)=4.98, p=0.0004 *	T(11)=0.504, p=0.623	T(11)=3.597, p=0.004 *	T(11)=3.038, p=0.011 *
Simple	T(11)=2.663, p=0.022 *	T(11)=1.514, p=0.158	T(11)=1.769, p=0.104	T(11)=4.883, p=0.0004 *	T(11)=5.075, p=0.0003 *
Control	T(11)=3.582, p=0.004 *	T(11)=3.178, p=0.008 *	T(11)=-0.271, p=0.791	T(11)=8.682, p=0.000002 *	T(11)=2.713, p=0.020 *

***Table 4.5**: Within-groups paired T-test results for components when heard in a single earcon and when as part of a compound earcon. Significant results are marked with an asterisk.*

Table 4.5 shows that, in the musical group, all the scores were significantly worse in phase IV except for the type and file scores where there was no significant difference. In the simple group all scores were worse except for family and file. In the control group only the file score was not significantly changed from the phase I. The fact that all the phase II scores were better than their phase IV equivalents confirmed the results from Table 4.3 where phase II was significantly better than phase IV. The results showed that the file scores were unchanged across phases I and IV. This was due to the generally low scores on file in both phases; the already low scores did not get any worse. As Figure 4.11 showed, file was the worst recognised component in phase IV in all three groups. This confirms the hypothesis that pitch alone is not good for recognition.

The overall results for phase IV showed that there were no differences between the three groups with combined earcons. The overall recognition rates of the combined earcons were the same as for phases I and III but phase II was better recognised. These results show that recognition rates did not fall when earcons were combined. The hypothesis that combined earcons would be as easy to recognise as the individual components was supported in the case. However, the overall level of recognition was still only 40% - 50% in phase IV. When the individual components of the phase IV earcons were tested against their equivalents in previous phases, menu and item recognition in phase IV were both significantly worse than in phase II. This confirmed the overall results where phase II had the best level of recognition. File scores were the lowest in all of the groups, again indicating that pitch alone is not a good method for differentiating earcons.

4.4.5 Musicians versus non-musicians

The overall recognition rates of earcons were low. A maximum of only 58% was reached as the best score overall. Perhaps this occurred because earcons are difficult to recognise and required musical skill? If earcons were only usable by trained musicians

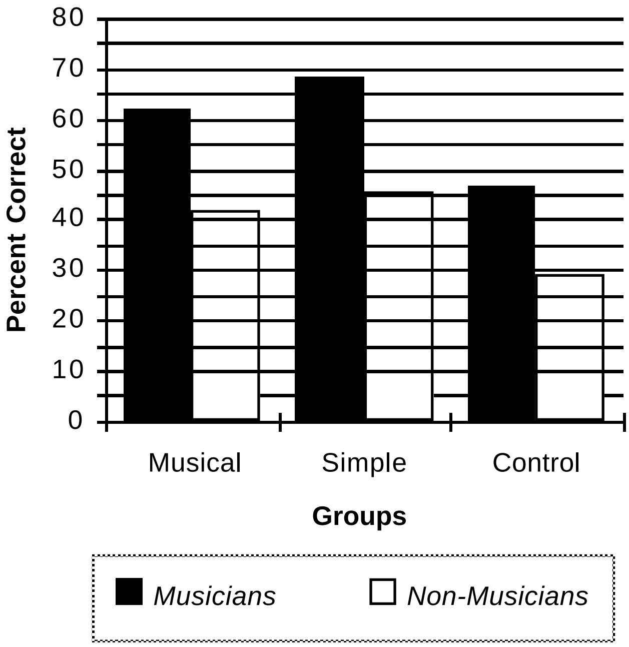


Figure 4.12: *Scores of musicians and non-musicians in phase IV of Experiment 1.*

then their effectiveness in the sonification of information would be limited. An investigation to find out if musicians were better at recognising earcons than non-musicians was conducted. The hypotheses suggest that musical ability would have no affect on scores. The raw results data for the following analysis are included in Appendix A Table 1. The results of the ANOVA tests for each phase are shown in Table 4.6.

Musical group

The earcons in the musical group from Experiment 1 were, overall, not statistically significantly better recognised by the musicians than the non-musicians (see Table 4.6). This meant that a non-musical user of a multimodal interface using earcons would have no more difficulties than a musician. The only time a significant difference occurred was in phase IV where detailed analysis showed that the musicians were better at identifying file. A between-groups (where the two groups were: Musicians and non-musicians) ANOVA showed an effect (F(1,10)=8, p=0.0179). This could have been due to musical training and skill allowing the identification of individual pitches more accurately. The results from phase IV are shown in Figure 4.12.

Simple group

There was a significant difference in overall scores in phases I and IV in the simple group (see Table 4.6). A more detailed between-groups ANOVA showed that the musicians were significantly better than the non-musicians with the phase I type and family (type: F(1,10)=10.7, p=0.0084, family: F(1,10)=5.12, p=0.04). This was repeated in phase IV where the musicians were again better on type and family (type: F(1,10)=9.59, p=0.04, family: F(1,10)=8.13, p=0.0172). The problems with the rhythms in phase I have been discussed above. The musicians were able to use the difficult rhythm information more effectively because of their greater musical training. In a similar manner the musicians were able to recognise the simple tone timbres that the non-musicians found harder to differentiate.

Control group

Overall there were no differences between the musicians and non-musicians in this group. In phase IV, however, the musicians did significantly better than the non-musicians with the menu and item. A more detailed between-groups ANOVA showed: menu F(1,10)=6.21, p=0.0319 and item F(1,10)=6.16, p=0.0324.

Musicians vs. non-musicians	Phase I	Phase II	Phase IV
Musical group	F(1,10)=0.34, p=0.571	F(1,10)=3.39, p=0.095	F(1,10)=3.72, p=0.082
Simple group	F(1,10)=13.42, p=0.0044 *	F(1,10)=0.55, p=0.476	F(1,10)=4.92, p=0.050 *
Control group	F(1,10)=0.19, p=0.671	F(1,10)=3.5, p=0.091	F(1,10)=3.35, p=0.0971

***Table 4.6**: ANOVA of musicians versus non-musicians in phases I, II and IV for each of the groups. Significant results are marked with an asterisk.*

These results indicated that if musical timbres were used for earcons then musically untrained subjects would not be at a disadvantage to musicians. This was in line with the work of Prior & Troup [137]. They showed that there were no differences between the two types of subjects when identifying timbres; musicians were no faster and made no less errors. They used musical timbres such as piano, violin and clarinet. When identifying rhythms again there was no difference in the number of errors but musicians were faster when sounds were presented to the right ear only. In the free-field set-up used for this experiment this is unlikely to confer much of an advantage. The only case where musicians proved to be better was with stimuli differentiated by pitch alone. This again indicated that pitch on its own was not good for differentiating earcons, especially if they were to be used by non-musicians. If simple tones or bursts of sound are used then musicians will have an advantage. Therefore, to create sounds that are usable by the general population musical earcons should be used.

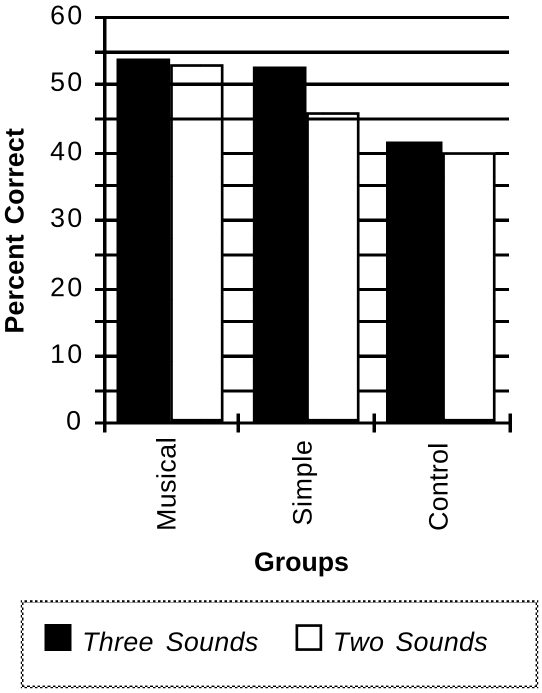


***Figure 4.13**: Comparing the recognition of three earcon compounds against two earcon compounds in Experiment 1.*

4.4.6 Three earcon compounds versus two earcon compounds in phase IV

One final factor was examined in this experiment: Would recognition rates fall if three compound earcons were played one after the other? Blattner *et al.* talk about making compounds of two earcons but there might be situations where more than two are needed for complex messages. The results described for phase IV above showed that compounds of two earcons could be recognised as well as individual earcons. One three-compound earcon was tested at the end of phase IV. The subjects were presented with two menu and one icon sound. The results for three earcon compounds were compared against the average percent correct scores from the recognition of the two earcon compounds. The results are shown in Figure 4.13 and raw data is given in Appendix A Table 3.

A two-factor repeated-measures ANOVA was performed on the data shown in Figure 4.13. It showed no main effect for group (F(2,33)=1.57, p=0.2238), no main effect for the number of earcons (F(1,33)=0.78, p=0.385) and no interaction between group and the number of earcons (F(1,33)=0.26, p=0.7732). The results show that there was no difference in the recognition rates of the earcons across the groups. There was also no difference in terms of the number of earcons presented which meant that the three earcon compounds were no more difficult to recognise than the two earcons ones. These results indicate that larger compounds than those suggested by Blattner *et al.* could be used without lowering recognition rates. It may be that the workload required to recognise three earcons could be higher but further experiments would be required to evaluate this.

4.4.7 Conclusions from Experiment 1

Some general conclusions can be drawn from this first experiment. The overall level of recognition of the earcons was not high: For example, the musical group only reached 58%. This result was achieved even though the experiment tested the 'worst-case' use of earcons. Earcons therefore did not seem to be an effective means of communicating information. The overall level of recognition in the musical and simple earcon groups was not significantly better than the control group which had less structured sounds. A more detailed analysis showed that there were differences between the groups.

Phase I showed that the musical timbres were significantly better than the simple tones proposed by Blattner *et al.* The rhythms used in this phase were ineffective. Subjects performed equally well without the rhythm information. Pitch was shown to be a poor method for differentiating earcons. The scores in phase II were significantly better than any other phase in the experiment. This was due to the significantly better recognition of rhythm in this phase. The musical and simple groups were significantly better than the control group (which used no rhythm). In this phase earcon recognition rose to approximately 75% for the simple and musical groups. Phase III showed that recognition of the phase I earcons was strong as it did not decrease after learning another set of stimuli. The combined earcons tested in phase IV were recognised at the same level as the individual earcons. This indicated that Blattner's method of combining them was effective. This was further reinforced when combined earcons with three components were tested and recognition rates again did not change. Finally, the results showed that musicians were no better at recognising earcons than non-musicians. This meant users did not have to be skilled musician to use earcons.

4.5 EXPERIMENT 2

From the results of the first experiment it was clear that the recognition of the icon sounds in phase I was low when compared to the menu sounds in phase II and this could be affecting the score in phase IV. The icon sounds, and especially the type (rhythm) information, needed to be improved along the lines of the menu sounds. It was decided that another experiment was needed to see if these problems could be overcome.

4.5.1 Sounds used

The sounds were redesigned so that there were more gross differences between each one. This involved creating new rhythms for files, folders and applications, each of which had a different number of notes. Each earcon was also given a more complex intra-earcon pitch structure. Figure 4.14 shows the new rhythms and pitch structures for folder, file and application. No changes were made to the rhythms and pitch structures used for the phase II menu item sounds (so their recognition rates were expected to stay the same).

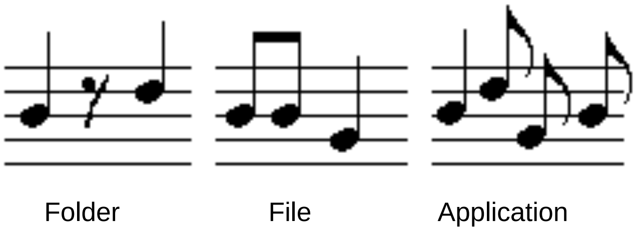


Figure 4.14: *New phase I rhythm and pitch structures used in Experiment 2.*

The use of timbre was also extended so that each family was given two timbres which would play simultaneously. The idea behind ‘multi-timbral’ earcons was to allow greater differences between families; when changing from one family to another two timbres would change not just one. This created some problems in the design of the new earcons as great care had to be taken when selecting two timbres to go together so that they did not mask one-another. Table 4.7 shows the timbres used in Experiment 2. The Draw family was given two Marimba timbres. This sounded much fuller and had more of an echo, making a more distinctive version of the single marimba.

Families	Timbres
Write	Piano/Electric Bass
Paint	Brass/Fantasy
Draw	Marimba/Marimba
HyperCard	Pan Pipes/Saxophone

Table 4.7: *Multi-timbral earcons used in Experiment 2.*

Findings from research into the psychoacoustics were included into the experiment. Patterson [128] (described in Chapter 3) gives some limits for pitch and intensity ranges. This led to a change in the use of register. In Experiment 1 all the icon sounds were based around C_3 (261Hz). All the sounds were now in put into a higher register, for example the folder sounds were put into C_1. In Experiment 1, the ‘file 2’ earcons were at G_5 (98Hz). This frequency was below the range suggested by Patterson. In Experiment 2 the ‘file 1’ earcons were played at C_1 (1046Hz) and the ‘file 2’s at C_3 (261Hz). These were now well within Patterson’s ranges.

In response to informal user comments from Experiment 1, a 0.1 second delay was inserted between the two earcons. Subjects had complained that they could not tell where one earcon stopped and the other started. Reich ([142], p 388) says pauses are "… crucial for the listener in enabling him to understand and keep pace with the utterance." They indicate the boundaries of chunks and provide vital processing time during the stimulus sequence. The pauses were added to see if this would give the subjects time to process the data as they heard it and improve performance.

4.5.2 Experimental design and procedure

The experiment was the same as the previous one in all phases but with the new sounds. A single group of a further twelve subjects was used. Subjects were chosen from the same population as before so that comparisons could be made with the previous results.

4.6 RESULTS AND DISCUSSION OF EXPERIMENT 2

As in Experiment 1 component data was collected for each of the subjects. The data was then added together to produce a new set of overall data. An overall analysis was performed to compare the scores of the different groups from both experiments. A more detailed analysis of the component data was then undertaken to compare the new group to the musical group from Experiment 1.

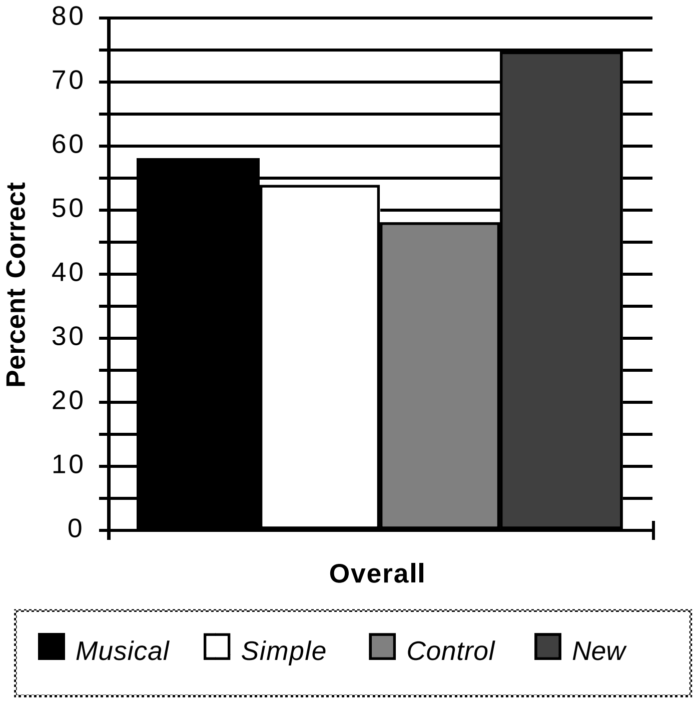


Figure 4.15: *Percentage of overall scores with Experiment 2.*

As can be seen from Figure 4.15, the new sounds performed much better than the previous ones. The new sounds reached 75% overall recognition rates as compared to

58% in the musical group from Experiment 1. A between-groups ANOVA on the overall scores indicated a significant effect (F(3,44)=6.169, p=0.0014). Scheffé F-tests showed that the new group was significantly better than the control group (F(3,44)=5.426, p=0.05) and the simple group (F(3,44)=3.613, p=0.05). This implied that the new earcons were more effective than the ones used in the first experiment. However, there was no significant difference between the new group and the musical group (F(3,44)=1.67, p=0.05). Figure 4.16 shows a comparison between the musical group (which was the best in all phases of Experiment 1) and the new group. Table 4.8 shows the ANOVA results between the musical group and new group. It can be seen that phases I and III were significantly improved. These results will be discussed in more detail below. The level of recognition in phases I and III was raised to close to that of phase II in the previous experiment. A more detailed investigation of the results was needed to see where the improvements over the previous experiment occurred. The raw results data are given in Appendix A Table 2.

The overall level of recognition of the new earcons was 75%. This indicated that the new earcons were an effective means of communicating information in sound. The problems of Experiment 1 had been overcome. The results show that with careful design effective earcons can be created.

A further analysis was carried out to find if any of the phases of the new group were significantly better recognised than any others. A one-factor ANOVA was carried out on the percentage data (as shown in Figure 4.16) for the new group. The results showed

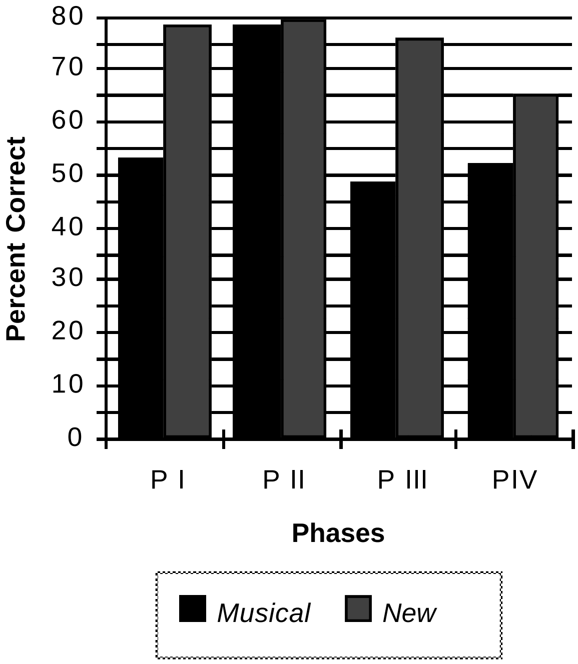


***Figure 4.16**: Breakdown of overall scores per phase for Experiment 2.*

that there were no differences due to phase (F(3,44)=1.039, p=0.384). This indicated that the performance in each of the phases was equivalent. This was an improvement over the results in Experiment I where, in the musical group, phase II was significantly better than all of the other phases. This showed that the advantages of phase II in Experiment 1 had been successfully applied to the other phases and recognition rates raised accordingly.

Phase I	Phase II	Phase III	Phase IV
F(1,22)=9.23, p=0.006 *	F(1,22)=0.03, p=0.8551	F(1,22)=5.85, p=0.0243 *	F(1,22)=3.67, p=0.0684

***Table 4.8**: Between-groups ANOVA results for the musical group from Experiment 1 and the new group from Experiment 2. Significant results are marked with an asterisk.*

4.6.1 Phase I

A more detailed analysis of each of the phases was carried out to find if any differences were hidden within the overall data. Table 4.8 shows that there was a significant improvement in the overall score in this phase as compared to the musical group from the previous experiment. Detailed ANOVA's were carried out to find what had caused the improvements. The overall recognition rate in phase I was increased because of a very significantly better type score (differentiated by rhythm) in the new group (F(1,22)=26.677, p=0.0001). The scores increased from 49.1% in the musical group of Experiment 1 to 86.6% in the new group (see Figure 4.17). This indicated that the new rhythms were effective and very easily recognised.

The wider register range used to differentiate the files made a significant improvement over the previous experiment (F(1,22)=4.829, p=0.0388). This indicated that using the higher pitches and greater differences in register made it easier for subjects to differentiate one from another. There was no significance between the family score of the two groups (F(1,22)=0.27, p=0.6166). This indicated that the multi-timbral earcons did not confer any improvement over standard single timbres.

The general improvement in recognition in phase I (and especially rhythm) brought the scores up to the level of the musical group in phase II of the previous experiment. This indicates that with more careful design of earcons recognition rates can be significantly improved.

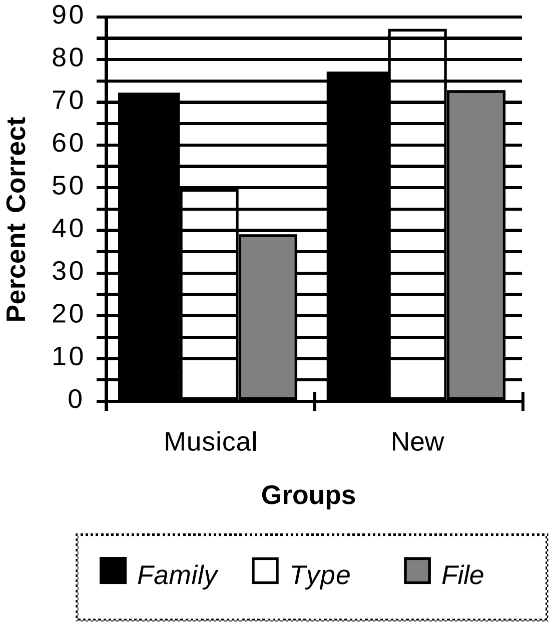


Figure 4.17: *Breakdown of scores for phase I of Experiment 2 compared to the musical group from Experiment 1.*

4.6.2 Phases II and III

The scores in phase II were not significantly different to the previous experiment, as was expected because similar earcons were used. Again, the multi-timbral earcons did not confer any advantage. In phase III the scores were not significantly different to phase I, again indicating that the sounds were easily remembered. A between-groups repeated-measure ANOVA was carried out on the phase I and III data. This showed an effect for group, as was expected but no effect for phase and no interaction (group: F(1,22)=7.89, p=0.0102), phase: F(1,22)=0.73, p=0.4014), interaction: F(1,22)=0.04, p=0.845)). This confirmed the overall results in that the new group was better than the musical group, but that there was no difference between phases I and III.

4.6.3 Phase IV

Table 4.8 shows that, in phase IV, the new group just failed to reach significance over the musical group at the 95% level. A detailed investigation of the components of phase IV was carried out by analysis of variance. The new earcons were significantly better than the musical ones from the previous experiment in terms of type (F(1,22)=9.135, p=0.0063) and family (F(1,22)=4.989, p=0.036). There was no significant difference in the file score (F(1,22)=1.06, p=0.3134). Figure 4.18 shows these results. The menu and item scores were not different, as was expected, because the same earcons were used as in Experiment 1. Type (differentiated by rhythm) was significantly improved, as it was in phase I. This again indicated that the new rhythms were more effective than the ones from the previous experiment. File scores rose from 34% to 50% but this difference

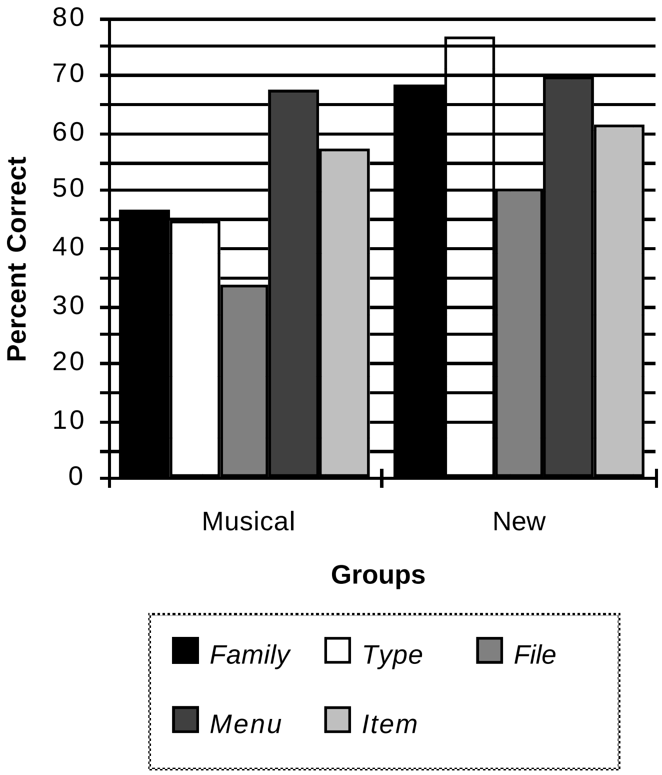


Figure 4.18: *Breakdown of scores for phase IV of Experiment 2.*

failed to reach significance. These results indicate that, even when bigger differences are used, pitch on its own may not be very reliable for absolute identification of earcons. It would best be used in combination with other parameters.

The multi-timbral earcons made no significant difference in phase I. The family score for the new group was not significantly different to the score in the musical group. There were also no differences in phases II or III. However, in phase IV the recognition of family was significantly better in the new group than in the musical group (F(1,22)=4.989, p=0.036). A further analysis of the data showed that there was no significant difference between the phase I and phase IV scores in the new group. However, the phase IV score for the musical group was worse than phase I (see Table 4.5). The multi-timbral earcons significantly improved recognition over the musical ones from the previous experiment. Their greater differences may have made them more distinguishable and memorable and so easier to recognise in the more complex compound earcons.

The results from phase IV showed that there were no differences between hearing the earcons individually and hearing them as part of a compound earcon. Both type and family recognition were significantly better than in the musical group from Experiment 1. Multi-timbral earcons were also shown to be effective at increasing the recognition of family in compound earcons.

4.6.4 Musicians and non-musicians

The results show that there was no significant difference in performance between the musicians and non-musicians with the new sounds in Experiment 2. A between-groups ANOVA was carried out and this showed no effect (F(1,10)=0.1, p=0.759). This indicated that the new musical earcons were the most effective way of communicating complex information for general users; they did not require any musical training in order for high recognition rates to be achieved. The raw data for these results are in Appendix A Table 2.

4.7 GUIDELINES

From the results of the two experiments and studies of literature on psychoacoustics some guidelines have been drawn up for use when creating earcons. These should be used along with the more general guidelines given in [163, 164] and described in Chapter 3. These guidelines form part of the structured method for adding sound to human-computer interfaces; they suggest what sounds should be used. A designer could use the guidelines to create earcons that could effectively communicate complex information in sound.

One overall result which came out of the work is that much larger differences than those suggested by Blattner *et al*. [25] must be used to ensure recognition. If there are only small, subtle changes between earcons then they are unlikely to be noticed by anyone but skilled musicians (if absolute judgements must be made).

- ❖ *Timbre*: Use musical instrument timbres. Where possible use timbres with multiple harmonics as this helps perception and can avoid masking. Timbres should be used that are subjectively easy to tell apart. For example, on a musical instrument synthesiser use ‘brass’ and ‘organ’ rather than ‘brass1’ and ‘brass2’. However, instruments that sound different in real life may not when played on a synthesiser, so care should be taken when choosing timbres. Using multiple timbres per earcon may confer advantages when using compound earcons

- ❖ *Pitch*: Do not use pitch on its own unless there are large differences between those used (see register below). Complex intra-earcon pitch structures are effective in differentiating earcons if used along with rhythm. Some suggested ranges for pitch are: Maximum: 5kHz (four octaves above C_3) and Minimum: 125Hz - 150Hz (the octave of C_4).

- ❖ *Register*: If listeners are to make absolute rather than relative judgements of earcons then pitch/register should not be used. A combination of pitch and another parameter would give better performance. If register alone must be used then there should be large differences between earcons but even then it might not be the most effective method. Two or three octaves difference should be used. This is not a problem if relative judgements are to be made.

- ❖ *Rhythm*: Make rhythms as different as possible. Putting different numbers of notes in each rhythm is very effective. Patterson [128] says that sounds are likely to be confused if the rhythms are similar even if there are large spectral differences. Small note lengths might not be noticed so do not use notes less than sixteenth notes or semi-quavers. This depends on the tempo. If 180 bpm is used then sixteenth notes last 0.0825 sec.

- ❖ *Intensity*: Although intensity was not examined in this test some suggested ranges (from Patterson) are: Maximum: 20dB above threshold and Minimum: 10dB above threshold. Care must be taken in the use of intensity. The overall sound level will be under the control of the user of the system. Earcons should all be kept within a close range so that if the user changes the volume of the system no sound will be lost (see Chapters 2 and 3).

- ❖ *Combinations*: When playing earcons one after another use a gap between them so that users can tell where one finishes and the other starts. A delay of 0.1 seconds is adequate. If the above guidelines are followed for each of the earcons to be combined then recognition rates should be similar to that of individual earcons.

4.7.1 The guidelines as part of the structured method for integrating sound into user interfaces

The research described in this chapter has shown that earcons are an effective method of communicating information in sound. Earcons can now be used to answer the question: What sounds should be used at the interface? The guidelines for the design of earcons form half of the structured method for integrating sound into human-computer interfaces. They allow a designer with no knowledge of sound design to create a set of earcons that will be effective. He/she can be sure that the earcons will be perceivable and recognisable by listeners because they incorporate knowledge of earcon design and psychoacoustics.

4.8 FUTURE WORK

This research did not test the speed of presentation of earcons. The single earcons took a maximum of 1.32 seconds to play and the combined ones a maximum of 2.64 seconds. In a real application they would need to be presented so that they could keep up with activity in the interface. A further experiment would be needed to test the maximum speed of presentation that could be achieved whilst retaining acceptable rates of recognition.

The subjects only heard each of the earcons three times in the training parts of the experiment but still reached 75% recognition rates in Experiment 2. A more long term study would show what levels of recognition could be reached when subjects had more time to learn the sounds.

4.9 CONCLUSIONS

The results from the two experiments described here indicate that if earcons are carefully designed they can be an effective means of communicating information in sound. The work described has experimentally demonstrated that earcons are better at presenting complex information than less structured bursts of sound. This gives a formal basis for their use in future systems. The results described were achieved even though the training and testing of the earcons was designed to be the ‘worst case’. So, as they proved to be effective here, they should be usable in systems where the training given is better.

The overall results from Experiment 1 were disappointing. Earcons were not shown to be capable of communicating complex information. A more detailed analysis did show some of the advantages of earcons and also that problems with recognition of rhythms were reducing overall scores. Experiment 2 showed that the initial problems could be overcome. The 75% recognition rates achieved in that part of the experiment demonstrated that earcons could be used to communicate structured information better than less structured sounds such as in the control group of Experiment 1.

Using musical timbres for earcons proved to be more effective than using the simple tones proposed by Blattner *et al.* [3]. Both the musical group and the new group obtained high recognition rates with musical timbres. Multi-timbral earcons were developed and shown to help recognition of compound earcons. Rhythm was revealed to be an important factor in the recognition of earcons in Experiment 2. In Experiment 1 the rhythms used were no better than pitch alone. In Experiment 2 they were redesigned and shown to be as good as timbre. Recognition of earcons based solely on pitch/register was shown to be poor even with the improvements made in Experiment 2. Absolute judgements are difficult for listeners without perfect pitch. If relative judgements are to be made then these problems will not occur. Overall recognition rates were maintained from phase I to the re-test in phase III. This gave some indication that memory for earcons is strong. Subjects remembered the phase I sounds after learning and being tested on the phase II sounds, which were very similar.

Experiment 2 showed that compound earcons could be used without any decrease in rates of recognition. This was a very important finding as it validated one of the main claims for earcons made by Blattner *et al.*: That earcons can be built-up into structured combinations. The work suggested that listeners were able to recognise three-earcon combinations without a decrement in performance. The two experiments also showed that musicians were no better at recognising earcons than non-musicians, if musical earcons were used. This is important as it allows earcons to be effective in interfaces used by the general population, rather than being restricted to use in interfaces for musicians.

The subtle transformations suggested by Blattner have been shown to be too small for accurate recognition by subjects and that gross differences must be used if differentiation is to occur. The results from Experiment 2 showed that high levels of recognition could be achieved by careful use of pitch, rhythm and timbre. A set of guidelines was proposed, based on the results of the experiments, to help a designer of earcons make sure that they will be easily recognisable by listeners. This forms half of the structured method for adding sound to human-computer interfaces.

The results of this research mean that there is now a strong experimental basis to prove earcons are effective if designed around the guidelines presented here. This work has shown that earcons can be individually recognised rather than recognition being based on hearing a relative change between two sounds. Earcons could therefore be used as landmarks in an auditory space where they give absolute information about events, for example. Developers can create sonifications of data or multimodal interfaces that use earcons safe in the knowledge that they are a good means of communication.

CHAPTER 5: PARALLEL EARCONS: REDUCING THE LENGTH OF AUDIO MESSAGES

5.1 INTRODUCTION

If non-speech audio feedback is to be used at the human-computer interface it must be able to keep pace with the interactions that occur. If it does not, and either the system has to wait for the sound to finish before continuing, or the sound playing refers to an interaction that has completed, then it will not be effective. It will not provide the user with any advantage so there will be no reason to use it. Sound takes place sequentially in time. One way to reduce the length of time a compound audio message takes, so that it can keep pace, is to play its sequential component parts in parallel. Phase IV in the previous chapter showed that the two component parts of a compound earcon could be played serially. This chapter suggests that these two parts could be played in parallel. An experiment is discussed that attempted to discover if this was an effective method of reducing the length of compound earcons. This work goes a step further in answering the question: What sounds should be used at the interface? It shows that earcons can be played at the speeds necessary to be effective at the human-computer interface. Some additions to the guidelines that are half of the structured method for adding sound to interfaces are proposed based on the work in this chapter. This work has been submitted for publication in [34].

5.1.1 The drawbacks of compound earcons

Interactions tend to happen quickly and audio feedback must be able to keep pace with them. The main drawback of compound, or *serial*, earcons as proposed by Blattner *et al.* [25] is that they can take a long time to play (1.3 - 2.6 seconds in the experiments described in Chapter 4). Each motive lasts a particular length of time depending on its notes and the tempo and these are then combined to produce longer compound earcons. Compound earcons could be played more rapidly (at a faster tempo) to overcome this problem but then errors in recognition may occur. The experiments described in Chapter 4 did not test the maximum speed of playback to find out at what point user's recognition of the earcons broke down.

One alternative method of overcoming this problem is to play the earcon at the same rate but pack the information more densely. This can be done by playing two earcons in *parallel* so that they only take the time of one to play. With parallel compound earcons the individual parameters can be left as they are for serial earcons but two sounds be played at the same time. Figure 5.1 gives an example of serial versus parallel compound earcons. This method has the advantage that the guidelines from Chapter 4 can still be

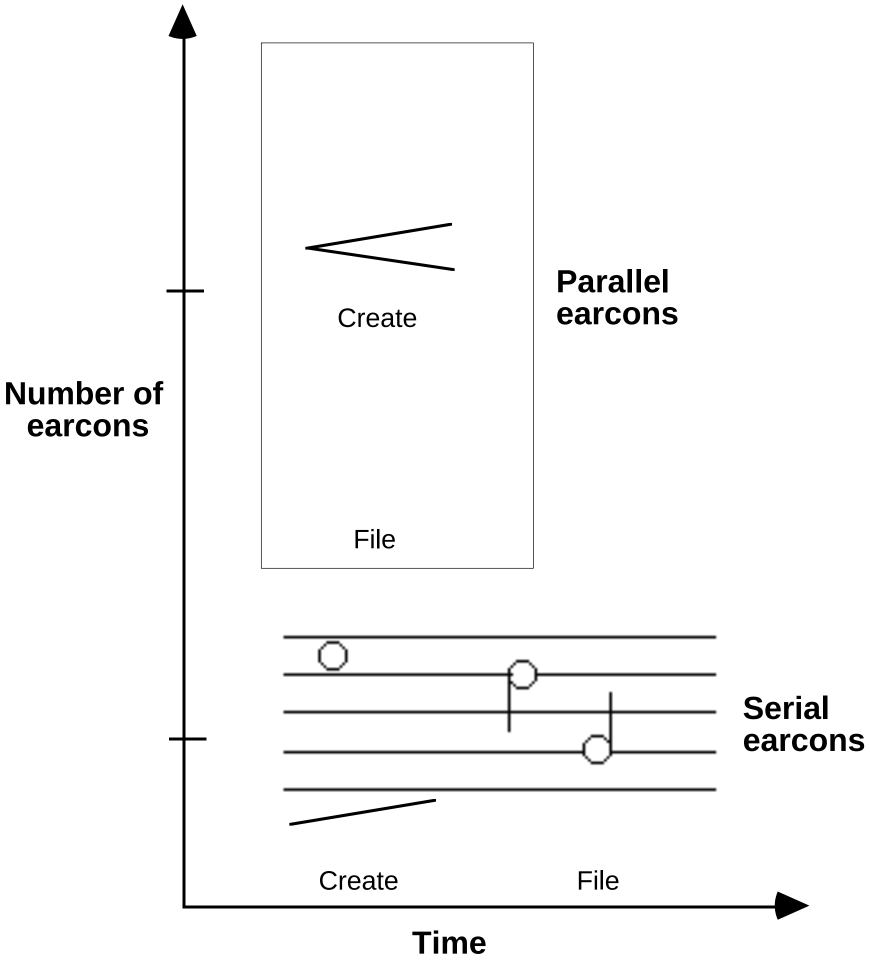


***Figure 5.1**: Serial and parallel compound earcons.*

applied but the disadvantage that it may be hard for the user to differentiate multiple sounds playing simultaneously. This chapter describes an experiment to investigate parallel earcons in more detail.

As Blattner, Papp & Glinert [26] say (p 448): “Our awareness and comprehension of the auditory world around us for the most part is done in parallel”. This suggests that parallel earcons could use a natural ability of the human auditory system. Research into auditory attention has investigated some of the problems of presenting multiple sounds simultaneously. Gerth [82] conducted several experiments to see if listeners could recognise changes in sounds when several were presented at once. As the density of sound (degree of polyphony or number of sounds playing simultaneously) increased recognition rates fell but remained at approximately the 90% correct level. Recognition rates did not fall significantly until three sounds were presented. The earcons discussed here are more complex than the sounds Gerth used so that combinations may be more difficult to recognise but only two are to be played in parallel. This may mean that two can be played in parallel without loss of recognition and perhaps four could be played if greater training was given. Blattner *et al.* [26] have begun to investigate parallel earcons to give information about maps.

Sonnenwald, Gopinath, Haberman, Keese & Myers [157] used parallel sound presentation to give feedback on parallel computations. They created a system, called InfoSound, that allowed the design of audio messages and their synchronisation with system events. In one example they describe how six part harmony was used to present six multiple concurrent processes. They do not describe any experiments to assess the effectiveness of the sounds nor whether listeners could extract information about the processes. The sounds Sonnenwald *et al.* describe are, again, simpler than earcons. One problem they describe is that designers found it difficult to add sounds as they were not trained in music composition. The work described in the previous chapter provided a set of guidelines to help interface designers without musical skills use sound. These guidelines will be enhanced to contain the results from the work described in this chapter.

Parallel earcons use some of the attributes of the musical theory of *counterpoint*. It is defined by Scholes ([148], pp 260-261) thus: “the combination of simultaneous voice-parts, each independent, but all conducing to a result of uniform coherent texture” (voice parts may include instrumental voices). In counterpoint individual instruments play separate musical lines which come together to make a musical whole. With parallel earcons, each component earcon is separate but the whole combined sound gives the meaning. This type of structure may give musicians an advantage over non-musicians that they never had with serial earcons.

5.1.2 The length of earcons

Blattner, Sumikawa & Greenberg [25] (p 28) suggest that "The optimal number of pitches in a motive is two to four". They state two reasons for this: Sounds take time to play so should be short; and with more than four pitches undesirable melodic implications can arise. They suggest (p 28): "Hearing a simple tune more than ten times a day potentially irritates users and could cause auditory fatigue". This paper argues that it is the *duration* of the earcons that is important, not the number of notes. Earcons can be kept to a short duration and still have more than four pitches by using shorter note lengths. One of the results from Chapter 4 was that gross differences between earcons made them more recognisable. If only four note lengths are used then there are not enough notes to create a sufficient variety of different rhythms. Blattner *et al.* [25] suggest that people become irritated by tunes low in complexity. A two-note earcon might avoid the melodic implications of a longer one and not be recognised as a tune but still be as annoying because of its low complexity. Think, for example, of a two-note alarm which even when heard at a low intensity can be annoying. It may be that earcons making up simple tunes will be more pleasing to the user and also easier to recognise as they can be heard as complete, well formed units; as Deutsch [52] suggests, the auditory system tries to make all groups of auditory stimuli into patterns and structures. In the experiment described here, the duration of the earcons was fixed at one second but within this up to six notes were used.

One factor that may give parallel earcons an advantage over serial ones is the *recency* effect [11], the term used to describe the enhanced recall of the most recently presented items. As serial earcons have to be held in memory for longer (because they take twice as long to play) the first part of the compound earcon might be forgotten and only the second, more recent earcon remembered. This problem would become more pronounced with longer earcons. This might have been causing the lower recognition rates in the phase IV in Chapter 4. Parallel earcons have both parts played at the same time so there is less time to forget any one earcon.

5.2 EXPERIMENT

An experiment was designed to see if the recall of parallel earcons was as accurate as the recall of serial earcons as shown in Chapter 4. The experiment was very similar to the one described in the previous chapter. However, there were only three phases: In the first phase the subjects learned earcons for objects (icons); in the second phase subjects learned earcons for actions (menus); in phase III subjects heard combined earcons made up of actions and objects (similar to phase IV in Chapter 4). Table 5.1 shows the format of the experiment. The work described in this chapter seeks to discover how well earcons can be recalled and recognised. It does not suggest that sounds should replace icons in the interface. Icons and menus were used as they provided a hierarchical structure that could be represented in sound.

Phases	Serial Group	Parallel Group
Phase I (train & test)	Object earcons	
Phase II (train & test)	Action earcons	
Phase III Presentation 1 (test only)	Serial compound earcons	Parallel compound earcons
Phase III Presentation 2 (test only)	Serial compound earcons	Parallel compound earcons

Table 5.1: *Format of the experiment.*

5.2.1 Subjects

Twenty-four subjects were used, half of them were musically trained. They were split into two groups of twelve, half of the subjects in each group being musicians. A subject was defined as being musically trained if she/he could play a musical instrument and read music. The subjects were undergraduate and postgraduate students from the University of York. None of the subjects used in the experiments described in the previous chapter were used again.

5.2.2 Sounds used

The sounds used were designed using the general guidelines put forward by Sumikawa [163] and the more specific ones from Chapter 4. Musical instrument timbres were used, as suggested by the results from the previous chapter. These are shown in Table 5.2. The phase I and some of the phase II rhythm, pitch and intensity structures are shown in Figure 5.2 and Figure 5.3. They have the gross differences suggested by Experiment 2 in the previous chapter. The sounds all lasted one second and were in 3/4 time. Greater care had to be taken in the timing of the earcons than in the experiments described in the previous chapter because two sounds would be played at the same time. They therefore had to be in time with each other, starting and finishing together, in order to sound pleasant.

The action earcons were in the scale of D Major and began at D_4 (146Hz). For example ‘Open’ was D_4, F_4, $A_{4,}$ the chord of D Major. The object earcons were in the scale of C Major and began at C_2 (523Hz). For example ‘File’ was C_2, C_2 and C_2, E_2, G_2, the chord of C Major. Two different scales were used to help listeners separate the earcons when they heard them together. The octave separation was similar to techniques in music where a bass line and a lead line are used. This helped overcome octave perception problems that occur where it can be difficult to differentiate the same note played in different octaves [50]. If different base notes are used then this problem is reduced. The notes D, F, A and C, E, G were chosen so they were distinct and also did not sound discordant when different combinations of the notes were played. To further help discrimination of sounds chords were used in the earcons for phase I but not phase II. Complex intensity structures were also used.

	Parallel & Serial Groups
Write	Piano
Paint	Brass
Spreadsheet	Pan Pipes
Menu 1	Marimba
Menu 2	Electric Organ
Menu 3	Cymbal

Table 5.2: *Timbres used in the experiment.*

As the sounds were to be played in parallel to one group, care had to be taken so that each of the earcons could be heard as a separate sound source or *stream*. If each earcon was not heard as a separate stream then they would mix together and neither of the earcons would be distinguishable. Bregman [30] and Williams have [183] put forward some principles which can be used to ensure that sounds are grouped into separate streams, see them for more on each of the principles mentioned below. These factors were first discussed in Chapter 2 but they are briefly mentioned here again in relation to the design of the earcons used in the experiment.

Similarity and dissimilarity

Components which share the same attributes will be perceived as related. All the pitches within an earcon were kept to the same octave. The action and object earcons were separated by two octaves to make sure the frequencies were dissimilar. To further increase the spectral dissimilarities, the object sounds used chords and the action sounds did not. Each of the earcons that could play together had a different timbre. This meant that the sounds had different spectral contents and amplitude and frequency modulations.

Proximity

Components which are close in time and frequency will be perceived as related. All the pitches used within an earcon were from the same octave so that there was frequency proximity. Chords also increased frequency proximity. Two octaves between the q= 0.33 seconds giving a tempo of 180 beats per minute (bpm).

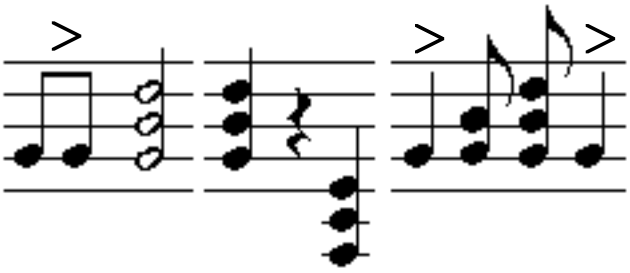


***Figure 5.2**: Rhythms, intensities and pitches used in phase I.*

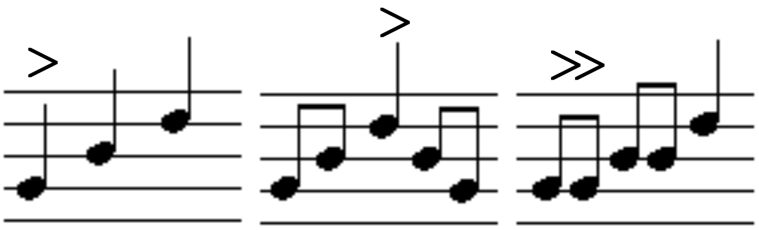


***Figure 5.3**: Rhythms, intensities and pitches used in phase II.*

actions and objects helped separate them into different streams because there was a large frequency distance.

Coherence

Components of streams change in coherent ways. All the components of a single earcon varied in terms of pitch and intensity together. These variations were distinct from those in the other earcon that might have been playing at the same time. The common modulations of amplitude and frequency due to the timbre of the different sounds playing helped to make each a different stream.

Spatial location

Components originating from the same spatial location will be perceived as related. Action sounds were presented on the right and object sounds on the left of the stereo space so that they would be heard as separate streams. This guideline is backed up by research from Mayfield (reported in [82]) where, without spatial separation of sources, recognition rates fell more rapidly as sound density increased than when there was stereo separation.

Rhythm

Rhythmic patterns tend to be perceived as sources [52]. Handel [88] suggests some methods for creating rhythmic groups. For the earcons used in the experiment two of these were used: *Intensity accentuation* (an accented note begins a group) and *Duration accentuation* (a long note ends a group).

All these principles were used in both the serial and parallel earcons as they help to differentiate any two sounds. The only difference with the serial earcons was that 0.1 second delay was placed between the two sounds (see Experiment 2 in the previous chapter) so that subjects could more easily tell where one finished and the other started [142].

The earcons for both groups were generated on a Roland D110 multi-timbral sound synthesiser and recorded by an Akai S950 digital sampler at a sampling rate of 48kHz. The sounds were all played through a Yamaha DMP 11 digital mixer controlled, using MIDI, by an Apple Macintosh computer and presented using external loudspeakers. In the experiments described in Chapter 4 the sounds were controlled directly from HyperCard, here HyperCard just played sound samples. HyperCard was not fast enough to generate all the sounds necessary for the parallel group in real-time, so sampling was used instead.

5.2.3 Experimental design and procedure

The design was very similar to that used in the previous chapter. However, some changes were made. In phase I here Write, Draw and Spreadsheet families were used. In the experiments in Chapter 4 some subjects commented that they had difficulty with the difference between ‘Draw’ and ‘Paint’ so it was decided to change ‘Draw’ to ‘Spreadsheet’ to avoid confusion. Another change made in this phase was that only three families were used instead of four and all of the members of all of the families were present. In Experiments 1 and 2, for example, all of the members of the paint family were present but only the HyperCard program was present in the HyperCard family. This new design would make the learning of the earcons more straightforward as subjects would not have to remember which members of each of the families were present. There were also no ‘File 2’ icons as there had been in the previous experiments. Those had shown that pitch alone was not a good method for differentiating earcons so there was no need to test it again here.

In phase II all the menus were three items long. This was changed from the previous experiments to avoid any indirect cueing that may have been given by different menu lengths. Phases I and II were identical for both groups of subjects. The purpose of these phases was to make sure the subjects would recognise the earcons when used in phase

III. In order to test the recognition of compound serial and parallel earcons in phase III any subject who did not reach a 65% recognition rate in both phase I and II was rejected. Only subjects who had learned the individual earcons could be tested on the combined ones. The serial earcons could then be compared to the parallel ones to see if recognition rates varied. Instructions were read from a prepared script.

The training of subjects on the timbres used was done in a more structured way than in the experiments described in Chapter 4. In their experiments, Corcoran, Carpenter, Webster & Woodhead [47] discovered that descriptions of timbres created by experts were the most effective for recall by subjects. Abstract labels or labels created by non-experts were less effective. In the previous experiments, subjects had to create their own names for the timbres. In this experiment, the experimenter (the expert) gave the names of the timbres used. A timbre was played and the subject told its name. This should have enabled subjects to build up a better mapping of timbre to family of menu. Subjects were scored in the same way as the previous experiments.

Phase I: Objects

Training: The subjects were presented with the screen shown in Figure 5.4. As in the previous chapter, subjects had to learn the names of all the icons. When they thought they had done this they wrote them down. If they were not correct they were allowed more time to learn them. This meant that, at the end of the training the subjects knew the names of all the icons present.

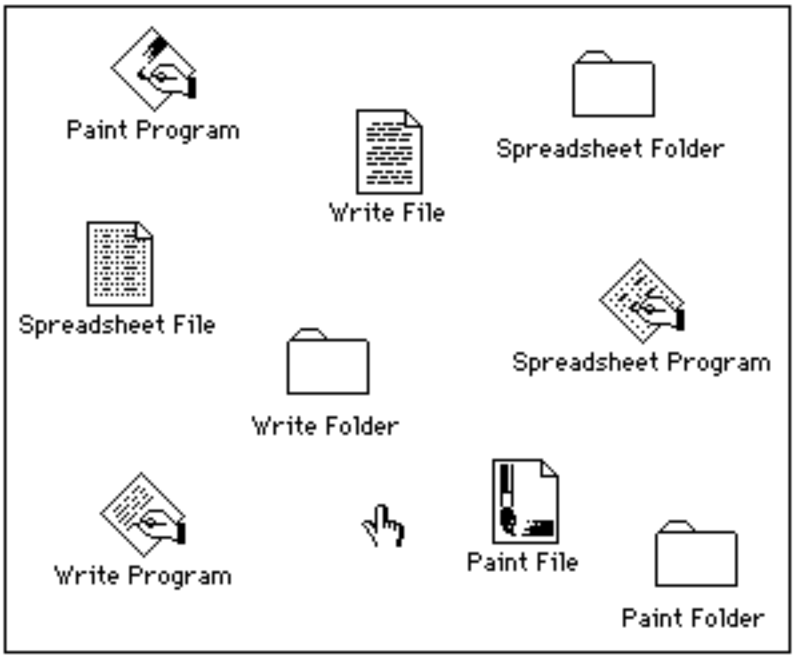


Figure 5.4: *Phase I object screen.*

Each of the objects on the display had a sound associated with it (the same as the previous chapter). The sounds were structured as follows. Each *family* of related items shared the same timbre. Items of the same *type* shared the same rhythm. The hierarchy of earcons used is shown in Figure 5.5. All of the information available graphically in the icons was available through sound in the earcons. The earcons were played one-at-a-time in random order to the subjects for them to learn and the whole set of sounds was played three times.

Testing: This was the same as in the previous chapter. During testing the screen was cleared and the earcons were played back in a random order. The subject had to supply what information he/she could remember about type and family. When scoring, a mark was given for each correct piece of information supplied (as in the previous chapter). Subjects were allowed to hear any stimulus a maximum of three times. Nine questions were asked. In the testing of this and the other phases subjects were not told the accuracy of their responses.

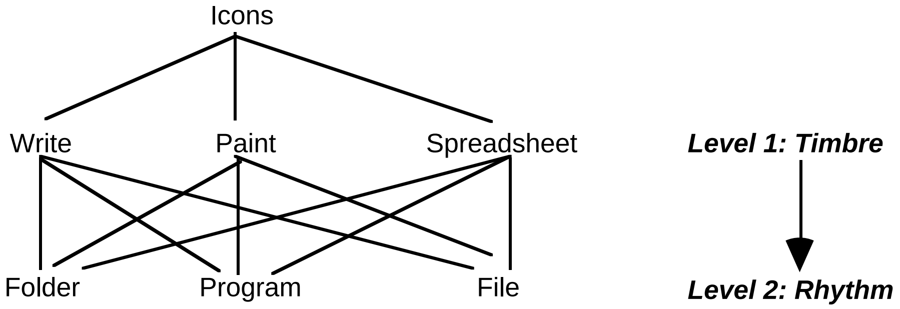


***Figure 5.5**: Hierarchy of earcons used in phase I.*

Phase II: Actions

In this phase earcons were created for actions. Each *menu* had its own timbre and the *items* on each menu were differentiated by an individual rhythm, pitch or intensity. The menus were not designed to represent any existing system such as the Macintosh or Windows menu structures. This was done so that no group of users would be favoured.

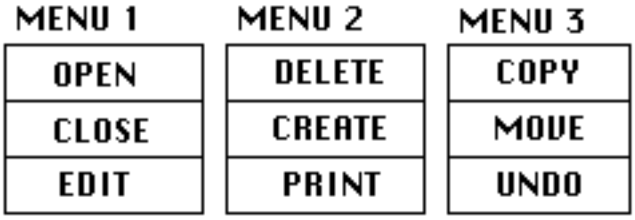


***Figure 5.6**: Phase II action screen.*

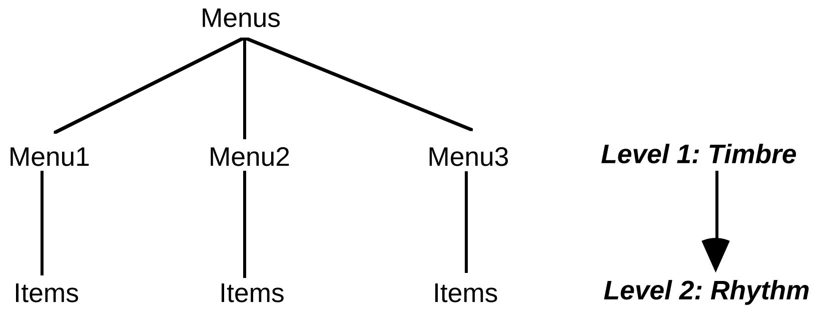


***Figure 5.7**: Hierarchy of earcons used in phase II.*

The screen shown to the users to learn the earcons is given in Figure 5.6 and the hierarchy of earcons used in Figure 5.7. The subjects were tested in the same way as before but this time had to supply information about menu and item. Subjects were allowed to hear any stimulus a maximum of three times. Nine questions were asked.

Phase III: Combinations

In the final phase subjects heard combined earcons. Phases I and II prepared the subjects for the main test in this phase. The parallel group heard parallel combined earcons and the serial group heard serial combined earcons made up of the sounds they heard in phases I and II. Before they were tested on phase III, subjects were presented with three examples of the type of combined earcons they were about to hear.

In this phase of the experiment an object earcon was always played with an action one. In the serial case an action sound was followed by an object one, in the parallel case an object and an action were played together. This was a more realistic method of presentation than that used in the previous experiments where the earcons were played in any order. In most graphical interfaces actions and objects are ordered. For example, in the Macintosh one might select some text (the object) and then choose to embolden it (the action). The previous experiments tested a ‘worst-case’ use of earcons; this experiment used them in a slightly more realistic way (although this experiment does not suggest that this is how sound should be used at the interface). Nine out of a possible set of 81 earcons were presented to the subjects during this phase. Each combined earcon was played once and the subject was then instructed to give all the information he/she could about the family, type, menu and item of the stimulus heard. The stimulus was then presented again and the subject could correct a previous answer or fill in any parts not recognised after the first presentation. This overcame a problem that occurred in phase IV of the experiments described previously, where subjects could hear the stimulus more than once before they gave their answer. This meant that subjects could listen to the first part of the earcon on the first presentation and the second on the subsequent presentation. The way the current experiment was designed forced the subjects to describe what they knew about the stimulus after one presentation. The second presentation was given to see what levels of recognition would be reached after greater exposure to the stimuli.

5.2.4 Experimental hypotheses

The experiment attempted to find out if compound parallel earcons were as recognisable as serial ones. The research in Chapter 4 showed that serial compound earcons could be recognised with 65% accuracy and also that musicians were no better than non-musicians, when musical instrument timbres were used. The main hypothesis for this experiment was that parallel earcons would be as recognisable as serial earcons and would thus reduce presentation time for audio messages. Listeners would not make more mistakes when listening to two complex stimuli. They could attend to and discriminate two complex sound sources at once because this is the way sounds are heard in the natural environment.

Testing that two results are equal (the null hypothesis) is dangerous because it may be that there really is a difference but the experiment is not sensitive enough to pick it up. The experimenter cannot tell if the scores are the same because there really is no difference or if the experimental design is not good enough to identify a difference that does exist. However, if high rates of recognition are achieved in the parallel earcon condition then that will show they are an effective means of communicating information in sound.

On the first presentation of the earcons, recognition rates would be lower than on the second presentation. Hearing the sounds for the first time would test initial recognition rates. The second presentation would give higher rates, similar to those of prolonged exposure. Musicians would, again, show no better performance than non-musicians. As described in the previous chapter, musical skill did not improve recognition of earcons. Musicians would not recognise more on the first presentation than the non-musicians because recognition of earcons is not dependent on musical skill. The overall recognition rates would be similar to those described in the previous chapter. The serial condition of this experiment was broadly the same as that described before so the results from this experiment would verify the results of the other.

5.3 RESULTS AND DISCUSSION

Figure 5.8 shows the overall scores for each phase in the experiment. The raw results data are given in Appendix B Table 1. For the data analysis the main area of interest was the differences between the groups. A two-factor repeated-measures ANOVA was carried out between the groups across each of the phases. The results showed there was no main effect for group (F(1,22)=0.07, p=0.801), there was a main effect for phase

(F(3,22)=10.45, p=0.0001) but no interaction between group and phase (F(3,22)=0.77, p=0.515). These results showed that there was no difference between the groups. This indicated that the parallel earcons were recognised as well as the serial ones confirming the main hypothesis of the experiment.

To find out where the main effect for phase occurred, post-ANOVA Tukey HSD tests were conducted for each group between each of the phases. The results showed that the only significant difference was between phase II and phase III(1) in both groups (Serial group II vs. III(1): Q(22)=4.71, p=0.05, Parallel group II vs. III(1): Q(22)=6.47, p=0.01). There were no other significant differences between the phases. Figure 5.8 shows that very high scores were achieved in phase II for both groups and the lowest scores for both groups were obtained in phase III(1).

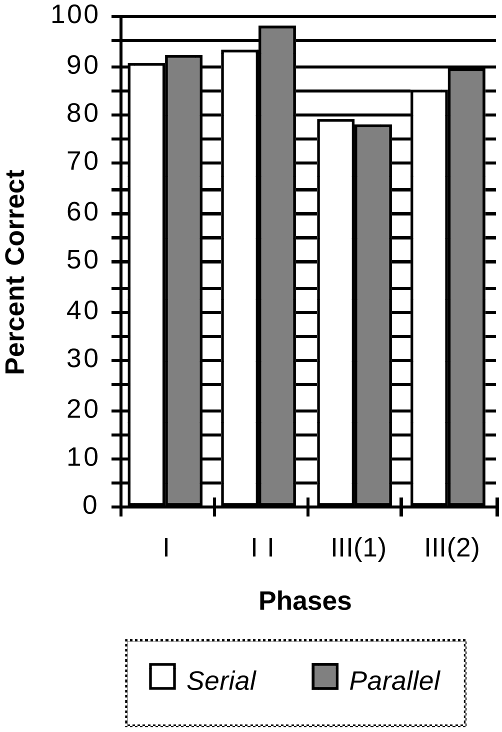


Figure 5.8: *Overall scores per phase.*

The recognition rates obtained here are broadly in-line with those of Experiment 2 in Chapter 4 where rates of approximately 80% were achieved for individual earcons and 65% for the compound earcons (as described above, the experiment here was simplified). The phase III(2) scores were very close to the scores of phases I and II. These results show that 90% recognition rates can easily be achieved with carefully designed earcons. This work gives further evidence to show that earcons are an effective means of communicating complex information in sound. As in the previous chapter, a more detailed analysis was undertaken to determine if the overall scores were hiding underlying differences between the groups.

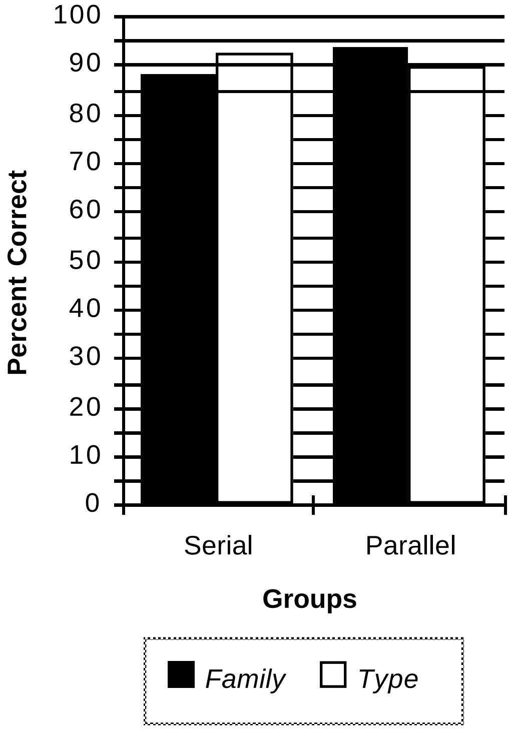


Figure 5.9*: Breakdown of scores for phase I.*

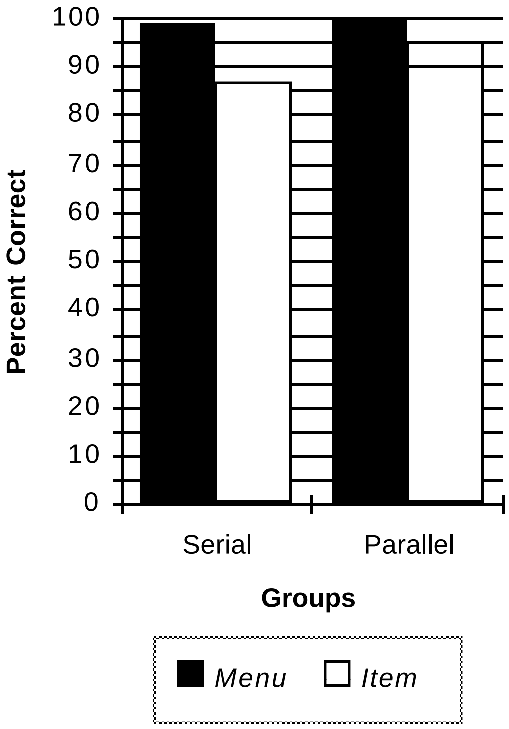


Figure 5.10*: Breakdown of score for phase II.*

5.3.1 Phases I and II

Figure 5.9 and Figure 5.10 show the scores for phases I and II. It was expected that there would be no difference between the groups in phases I and II as they both received the same stimuli. A one-factor ANOVA was conducted on the component data for each group in phases I and II. The results are shown in Table 5.3. There were no significant differences between the groups on any of the components.

Family	Type	Menu	Item
F(1,22)=1.19, p=0.288	F(1,22)=0.3, p=0.59	F(1,22)=1, p=0.328	F(1,22)=3.61, p=0.07

Table 5.3: *ANOVA analysis of serial versus parallel group scores in phases I and II.*

The high rates of recognition achieved in these two training phases meant that the subjects were well prepared for the main test in phase III. As mentioned above, any subject who did not reach 65% in any part of phase I or II was rejected. This led to the rejection of three subjects. See Section 5.3.5 on rejected subjects for more details.

5.3.2 Overall phase III(1) and III(2) results

In phase III the same stimuli were presented to the subjects twice to investigate what would happen to the recognition rates. The stimuli were ones that the subjects had been trained on in phases I and II. The first presentation was called III(1) and the second III(2). A two-factor repeated-measures ANOVA was carried out between the groups across the two presentations on the overall data. It showed no main effect for group (F(1,22)=0.08, p=0.777). This showed that there was no difference in recognition rates between the serial and parallel earcons. It showed a very strong main effect for the repeated measure from presentation III(1) to III(2) (F(1,22)=87.27, p=0.0001). This indicated there was a significant increase in recognition from III(1) to III(2). There was also a significant interaction between group and presentation (F(1,22)=6.04, p=0.022).

To find out where the main effect for the repeated-measure occurred Tukey HSD tests were conducted on the data for each group across presentations. The results showed that in both groups phase III(2) was significantly better than III(1) (serial group: Q(22)=6.902, p=0.01, parallel group: Q(22)=11.799, p=0.01). This difference can be observed in Figure 5.8. These results show that the subjects got significantly better when they heard the sounds a second time. This was investigated further to see where the interaction occurred. The differences between the III(1) and III(2) scores for both groups were calculated to see which group had increased the most. This was done by taking the phase III(1) score from the phase III(2) score for each group. A one-factor ANOVA was then used on these difference data. It indicated that the parallel group increased significantly more than the serial group (F(1,22)=6.04, p=0.0223). This can be seen in Figure 5.8.

Discussion

These results show that compound parallel earcons are as capable as compound serial earcons at communicating information. The recognition rates of both groups were not significantly different. This indicated that parallel earcons were an effective means of reducing the length of compound earcons without compromising recognition rates. Recognition rates were lower on the first presentation of the earcons, as was expected, but were still around 75%. On the second presentation, rates increased significantly to between 85% and 90%. This indicated that the more earcons were heard the better the recognition rates would be. This would be the situation if earcons were used in human-computer interfaces. The parallel group increased significantly more than the serial group from the first presentation to the second. However, as there were no overall differences in terms of group, this increase does not indicate that parallel earcons are more easily recognised.

The first presentation of the compound earcons was significantly worse than the recognition in phase II but by the second presentation there were no differences in recognition. This showed that the recognition rates of the combined earcons were as good as when the component earcons were heard individually.

5.3.3 Detailed phase III results

A detailed examination of phase III was undertaken to investigate recognition of individual components of the earcons. The data are shown in Figure 5.11. A two-factor repeated-measures ANOVA was conducted on the two groups across the eight components of the two presentations. As expected from the results described above, there were no differences in recognition of the components in terms of groups (F(1,22)=0.08, p=0.7776). This showed that, for each of the components in both presentations, the groups did not differ significantly in recognition rates. There was a significant difference in terms of the components (the repeated measure) (F(7,22)=9.33, p=0.0001) but no interaction between group and components (F(7,22)=1.16, p=0.3272).

A significant difference in terms of repeated-measure was expected as the overall results showed that presentation III(1) was significantly worse than III(2). Tukey HSD tests were conducted to find out where the differences occurred. The significant results are shown in Table 5.4 and Table 5.5, any results not shown were not significant. For the serial group it can be seen that the menu scores in both presentations were significantly better than any of the other components. Menu in presentation III(2) was significantly better than item, file and type in presentation III(1). Menu in presentation

III(1) was significantly better than item and family in the same presentation. There was no significant difference between the menu scores in III(1) and III(2). Figure 5.11 shows the high menu scores. The results for this group were similar to those in phase II versus phase I. There, menu was the best recognised of the components. Menu, differentiated by timbre, was a very powerful cue for the serial group. This, once again, confounds the definition of earcons given by Blattner *et al.* [25] where it is suggested that pitch and rhythm are the most important factors for recognition.

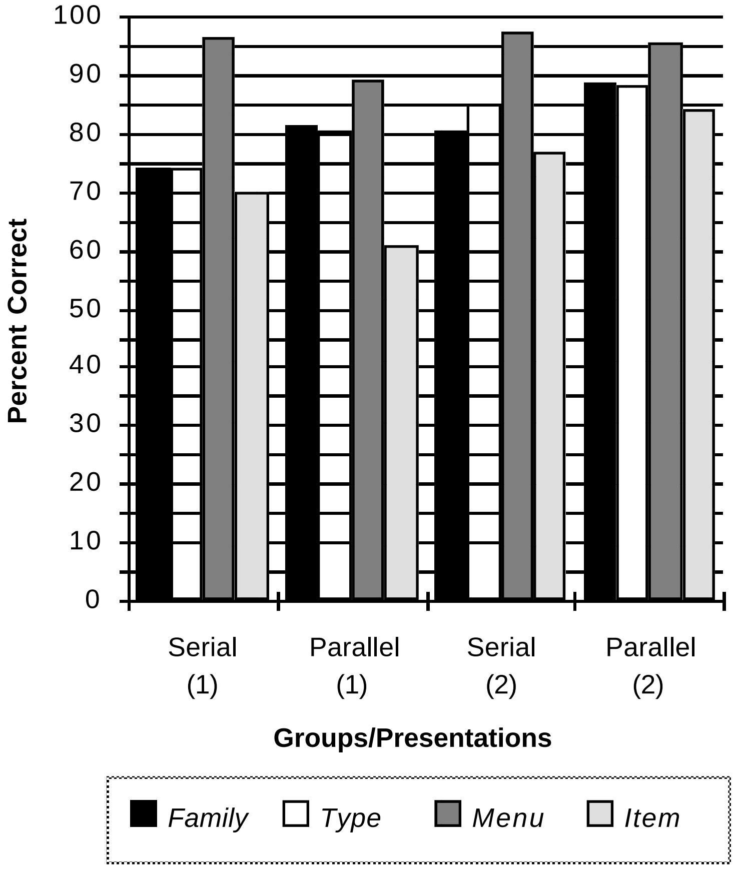


Figure 5.11*: Breakdown of scores for phase III presentations 1 and 2.*

In the parallel group a poor item score in phase III(1) accounted for the main differences between the presentations (see Table 5.5). All of the phase III(2) components were better than III(1) item. Menu in III(1) was also better than item. It may be that when two earcons are heard in parallel the rhythms are harder to detect. Rhythm is a complex component and, if listeners were to switch their attention from it to listen to the timbre, for example, information could easily be lost. However, in the second presentation item score increased significantly over the first presentation. There were no longer any differences between the item score and any other component of III(2). Therefore, with greater exposure to earcons (as would occur if they were being used in a human-computer interface) problems of rhythm recognition would not be an issue.

Menu2	Menu1
Menu2 vs. Item1 Q(22)=6.114, p=0.01	Menu1 vs. Item1 Q(22)=5.684, p=0.05
Menu2 vs. Family1 Q(22)=5.255, p=0.05	Menu1 vs. Family1 Q(22)=4.825, p=0.05
Menu2 vs. Type1 Q(22)=5.052, p=0.05	

Table 5.4*: Serial group Tukey HSD tests showing significant differences between the components in phases III(1) and III(2). Menu1 = menu score in III(1), Menu2 = menu score in III(2).*

Menu2	Family2	Menu1	Type2	Item2
Menu2 vs. Item1 Q(22)=7.781, p=0.01	Family2 vs. Item1 Q(22)=6.316, p=0.01	Menu1 vs. Item1 Q(22)=6.316, p=0.01	Type2 vs. Item1 Q(22)=6.114, p=0.01	Item2 vs. Item1 Q(22)=5.25, p=0.05

Table 5.5*: Parallel group Tukey HSD tests showing significant differences between the components in phases III(1) and III(2). Menu1 = menu score in III(1), Menu2 = menu score in III(2).*

Discussion

Looking at performance on the individual components of phase III the overall results are confirmed. The second presentations of the earcons were better recognised than the first and there were no differences between the groups on any of the components. This again indicates that parallel earcons are as effective as serial ones at communicating information.

The scores show that both groups had problems identifying the item component. This led to the lowest score in phase III. Although by the second presentation the score in the parallel group was nearly 85%. It could be that type (the equivalent component to item in phase I) fared better than item as subjects were able to use the structure information in the rhythms more effectively. There was less structure information available in the item sounds so that they were harder to remember. The item scores did increase significantly on the second presentation so, again, with practice subjects can reach high levels of recognition. As the results described above suggest, the scores for item were not significantly worse than type (both of these were based on rhythm).

5.3.4 Musicians and non-musicians

In Chapter 4 it was shown that, for combined serial earcons, the performance of musicians was not significantly better than non-musicians. In this experiment musicians were again compared to non-musicians to see if they performed better with combined parallel earcons. Comparing subjects in the serial group would also allow the general result from Chapter 4 to be confirmed. It may be that parallel earcons were more easily recognised by musicians as they were used to listening to complex sounds playing in parallel. They might also have obtained higher scores on the first presentation due to their greater skill. Figure 5.12 shows the overall scores of the musicians and non-musicians across all phases. The raw results data are given in Appendix B Table 1.

Results

The two groups were divided into four: Serial musicians, parallel musicians, serial non-musicians and parallel non-musicians. An overall two-factor repeated-measure ANOVA was performed on percentage data (as shown in Figure 5.12) between the groups across the phases. It showed no main effect for group (F(3,20)=0.65, p=0.5925), it showed a main effect for phase (F(3,20)=10.64, p=0.0001) but no interaction between group and phase (F(3,20)=1.07, p=0.3961). This showed that there were no differences between the musicians and non-musicians in any group on any of the phases. Therefore, musicians were not significantly better than non-musicians with parallel earcons.

To investigate where the main effect for phase occurred Tukey HSD tests were used within groups across phases. These tests showed that there were no differences between any of the phases in the serial musicians group. Serial non-musicians were significantly better in phase II than either III(1) or III(2) (II vs. III(1): Q(20)=5.531, p=0.01, II vs. III(2): Q(20)=3.921, p=0.05). Parallel musicians were better in phase II than III(1) (Q(20)=4.119, p=0.05) as were the parallel non-musicians (Q(20)=5.100, p=0.01). These results confirm those of the overall analysis where phase II was significantly better than III(1). From these results the parallel non-musicians performed better than the serial ones because they managed to increase recognition in III(2) to the level of individual earcon recognition.

These results confirm those in Chapter 4: There were no differences between musicians and non-musicians in the recognition of serial earcons. There were also no differences in recognition of parallel earcons. Musicians did not perform any better than non-musicians even with the more complex stimuli. The parallel group musicians did not reach higher recognition levels on the first presentation of the phase III earcons, as might have been expected due to their training. This seems to suggest that parallel earcons can be used effectively by those not skilled in music.

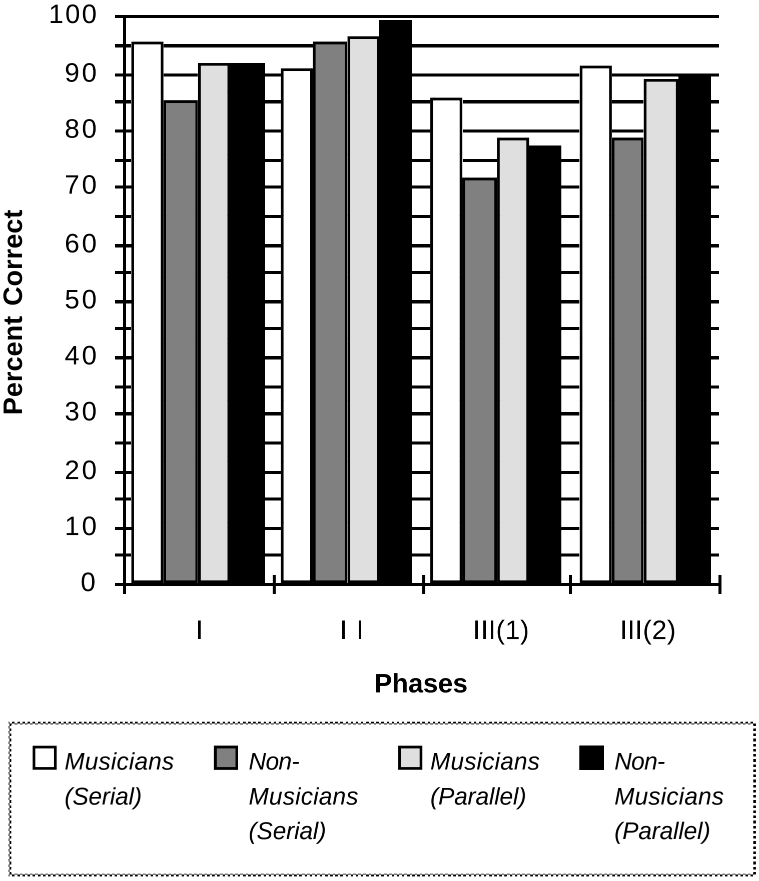


Figure 5.12*: Overall scores for musicians and non-musicians.*

5.3.5 Rejected subjects

As mentioned above, subjects were rejected if they reached below 65% correct scores in either phase I or II after hearing each of the earcons three times in training. Nine questions were asked in each of phases I and II so any subject who got a score of less than 5.85 was rejected. Subjects had to be able to recognise the individual component earcons before they could be tested on the combined ones. This lead to the rejection of three subjects who did not reach the required level. These subjects were all non-musicians and they all failed on phase II. The subjects rejected obtained scores of 4, 4 and 5 in this part of the test (the raw results for the rejected subjects are given in Appendix B Table 1). In phase II there was less structure information to help remember the sounds than in phase I which may have made it a harder test. In order to find out how much extra training each would need to reach the 65% level they were trained further. This involved going through the training and testing of the phase where the subject fell below the required level until they reached it. One of the subjects reached the required level after one further training session, one subject required two sessions and one subject never reached the required level, even after three more training sessions. This seems to indicate that some users may have problems with earcons. Some may require more training which could be done when the user initially came to the computer with an auditory interface. Some users may always have problems and this may be analogous to colour-blind users with coloured graphical interfaces.

5.4 GENERAL DISCUSSION

The results of phases I and II confirm those of Chapter 4 in that subjects were able to recognise the individual earcons with a great degree of accuracy. The high rates of recognition of timbre (up to almost 100% in phase II) again indicate that musical instrument timbres are very effective and users can easily identify them. The phase III results show that parallel compound earcons are as easily recognised as serial compound earcons. This means that parallel earcons are more effective in an auditory human-computer interface as they take only half the time to present to the user.

After a prolonged exposure to the parallel earcons it would be hoped that the subjects would hear the two separate earcons as a single ‘whole’ earcon. For example, earcons for ‘open’ and ‘write file’ would coalesce and be heard as an earcon for ‘open write file’. Listeners would become accustomed to the sounds and come to recognise the overall earcon. In order to test this another experiment would be needed which would train and test subjects with much longer exposure to the earcons.

5.4.1 Parameters for manipulating earcons

There are five parameters that Blattner *et al.* [25] propose can be manipulated to differentiate earcons. They suggest that rhythm and pitch are the primary (fixed) parameters and timbre, intensity (dynamics) and register are the secondary (variable) parameters.

Primary Parameters	Secondary Parameters
Rhythm	Pitch
Timbre	Intensity
Register	Stereo position
	Chords
	Effects (Echo, Chorus, Etc.)

Table 5.6: *The new parameters for manipulating earcons.*

The results from Chapter 4 indicated that timbre was much more important than suggested by Blattner *et al.* [25]. In that experiment (and this one) timbre was used to denote families of icons or menu items. The results showed that it played a much bigger role than that suggested by Blattner *et al.* [25]. Chapter 4 also showed that earcons differentiated by pitch alone were very difficult to discriminate. Register was shown to be more important so that big differences between earcons could be created. The earcons used in this current experiment were designed around the guidelines put forward in Chapter 4 and again high recognition rates were reached. It can therefore be argued that timbre and register, along with rhythm, are the primary parameters for creating the basic structure of a set of earcons. Secondary parameters, such as pitch, intensity, stereo position, chords and effects (such as echo or chorus) work together to help differentiate the earcons from each other (Table 5.6 shows these parameters, the items in each column of the table are not ordered).

5.4.2 Auditory streaming and earcon guidelines

The auditory streaming techniques, described above, that were used to differentiate the earcons proved to be effective. None of the subjects in the parallel group complained they were unable to separate the two earcons. These techniques were also effective on the serial earcons. The guidelines from the previous chapter fit well with the general principles of auditory streaming. These principles suggest ways to make components in the same stream similar and dissimilar to components of other streams. The earcon guidelines also try to do this so that there are big enough differences between earcons that subjects can recognise them individually.

The earcon guidelines, and the experiment described in this paper which used them, heavily stress *similarity* and *dissimilarity*. Each earcon family had a separate timbre so that items in the same family shared the same attributes. Chords were also used to give the object sounds different attributes to the action sounds. *Proximity* was also stressed in the guidelines. Related items were in the same octave and unrelated ones separated by two or more octaves. *Coherence* was important as each earcon had a different timbre with the attendant differences in modulation that brought about. *Spatial location* was not in the original guidelines but is another important method of differentiating earcons. Two locations were used in this experiment and a third central location could easily be added. MIDI allows at least 16 different stereo positions but it is unclear if each of these could easily be distinguished as the differences between them would be small. The work of Wenzel, Wightman, & Foster [177] (discussed in greater detail in Chapter 2) has shown that three dimensions are possible in synthesised sound, and this would provide more opportunities for recognisable locations. The methods described for creating rhythmic groups are an important addition to the guidelines as they make each of the rhythms into a whole and complete unit with a more defined start and end point.

The original earcon guidelines from the previous chapter have been shown to share some of the principles developed in auditory steam segregation research. This gives them a stronger foundation as auditory streaming research deals with some similar problems. Extensions have also been put forward to the guidelines to include spatial location and extended use of rhythm to create more complete rhythmic units. The guidelines now allow an interface designer to generate earcons that can be heard in parallel.

5.4.3 The new guidelines as part of the structured method for integrating sound into user interfaces

This research concludes the work in this thesis that attempts to answer the question of what sounds should be used at the interface. Further investigations of earcons would provide more complete guidelines but from the research here and in the previous chapter a workable set of guidelines has been developed. It could be used by an interface designer (who has no detailed knowledge of sound design) to create earcons that would be effective at communicating information. The earcons produced could be played in parallel if necessary to keep pace with interactions. These guidelines complete half of the structured method for integrating sound into human-computer interfaces.

5.5 FUTURE WORK

The next step for this work would be to find out the maximum number of earcons that could be recognised in parallel. It may be that more can be recognised (as Gerth [82] suggests) but that bigger differences between them would be required and that larger and larger amounts of training needed to reach equivalent recognition rates.

One other aspect to consider is the workload required to recognise the parallel earcons. The parallel earcons might require more effort to recognise but this might not affect performance in low workload situations (such as the experiment described here). When other tasks are being performed and more cognitive resources are in use then recognition rates might fall (see [92] for more on this). Further work to investigate the workload of parallel earcons as compared to serial earcons is needed. A similar experiment could be run again but workload measures recorded to see if the parallel earcons affected workload. See Chapter 7 and [35] for an example of this kind of evaluation of sonically-enhanced widgets using NASA Task Load Index workload measures [122]. If more earcons could be recognised in parallel then they could be used to give information about multiple processes running on a machine, for example. Each process might have its own timbre and spatial location to identify it and would play in the background. A subject could listen to any one of the earcons to tell the status of that particular process.

5.6 CONCLUSIONS

In Chapter 4 the results showed that earcons could be recognised with a high degree of accuracy. A problem still remained that earcons took too much time to play and, if they were to be used in human-computer interfaces, might not be able to keep up with the pace of interaction. Slowing interactions down so that the sounds could keep up would be unacceptable. The experiment described here has shown that the length of compound earcons can be reduced to half by playing them in parallel and the rates of recognition maintained. This means that displaying complex information in sound that can keep pace with interactions is possible using parallel earcons. This research will allow a wider application of sound at the interface.

Combined parallel earcons were shown to be as effective as the combined serial earcons proposed by Blattner *et al.* [25]. The results from this chapter confirm those of the previous one. The results are broadly the same, even though there were some differences in the experimental design. Results from the first presentation of the earcons, phase III(1), showed that on a single presentation almost 80% correct scores could be achieved. The second presentation showed that recognition increased significantly when subjects heard the sounds again. This indicated that, if earcons were used at the human-computer interface, then regular exposure would quickly lead to high levels of recognition.

Musicians have again been shown to be no better than non-musicians. This means that auditory interfaces will be usable by most users whatever their level of musical skill. The results of the re-training of rejected subjects, however, showed that some users may always have problems using sound. This is similar to a colour-blind person using an interface that depends heavily on colour. Some extensions have been put forward to the guidelines described in Chapter 4 based on research into auditory stream segregation. This more complete set of guidelines answers the question of where to use sound at the interface and forms half of the structured method discussed previously. By using the guidelines a designer could create effective sounds for an interface.

This work has extended that described in Chapter 4 and shown that earcons are not only an effective means of communicating complex information in sound but that they can do it at a rate which can keep up with the pace of interaction in an interface. Humans process multiple sounds in parallel in their everyday world and earcons are able to exploit this to overcome rate of presentation problems at the interface. Earcons have now been shown to be a very effective method of communicating information in non-speech sound.

CHAPTER 6: WHERE TO USE SOUND AT THE HUMAN-COMPUTER INTERFACE

6.1 INTRODUCTION

The previous two chapters showed that complex information could be communicated in non-speech sound using earcons. The research they described answered the question: What sounds should be used at the interface? The guidelines produced form part of the structured technique for integrating sound into user interfaces. However, this is only one half of the problem. Unless these sounds supply information that users need to know they will serve no real purpose; they will become an annoying intrusion that users will want to turn off. Therefore one other important question that must be answered is: Where should non-speech sound be used in the graphical human-computer interface? The answer to this will form the other half of the structured method for integrating sound into user interfaces.

The combination of graphical and auditory information at the interface is a natural step. In everyday life both senses combine to give complementary information about the world. The advantages offered by the different senses can be brought to the human-computer interface. At the present time almost all information presented by computers is through the visual channel. A multimodal interface that integrated information output to both senses could capitalise on the interdependence between them and present information in the most efficient way possible.

How then should the information be apportioned to each of the senses? Sounds can do more than simply indicate errors or supply redundant feedback for what is already available on the graphical display. They should be used to present information that is not currently displayed (give more information) or present existing information in a more effective way so that users can deal with it more efficiently.

Alty & McCartney [4] have begun to consider this problem in process control environments. They wanted to create a multimedia process control system that would choose the appropriate modality (from those available) for presenting information to the plant operator. A resource manager is used to choose in which modality feedback should be presented. They say (p 10):

> “The Resource Manager’s goal is not simply to allocate resources to ensure that all interactions may run, but to allocate the various media and mode resources in order to maximise user comprehension of the interaction”.

In such a system there is much information that must be presented and the appropriate method may not always be available because, for example, it is being used for other output at the same time. Alty & McCartney suggest that alternative media could then be used. Unfortunately, at the present time there is no method for deciding which is the best way to present the information, they say (p 10):

> “However, almost nothing is known about what constitutes the successful use of multiple media and modes of communication with the user. Hence, it is extremely difficult to specify any heuristics or constraints which may be successfully employed to drive a resource management system”.

The abilities of the resource manager were limited for this reason. In order to get around these problems, an interface designer using the system would design a dialogue and supply a set of alternative media that could be used to present the information him/herself. At run-time, the resource manager would choose the appropriate form of output based on simpler constraints. Alty & McCartney suggested that further research was needed on rules for combining media and modes before the system could operate without the designer specifying what was needed. The research described in this chapter aims to deal with some of these problems by providing an analysis technique that suggests where sound could effectively be used in the interface.

Sound is often used in an *ad hoc* way by individual designers. Gaver [74] used a more principled approach in the SonicFinder that used sounds in ways suggested by the natural environment (see Chapter 3 for more detail). However, in an interface there are many situations where there are no natural equivalents in the everyday world. The SonicFinder also used sound redundantly with graphics. This proved to be effective but it is suggested here that sound can do more. A method is needed to find situations in the interface where sound might be useful but prior to this work there was no such method. It should provide for a clear, consistent and effective use of non-speech audio across the interface. Designers will then have a technique for identifying where sound would be useful and for using it in a more structured way rather than it just being an *ad hoc* decision.

Sound could be added to the interface in many different ways. A top-down approach might be taken: An overall example system could be created, such as a word processor, that used sound to provide information to the user. This large scale example would show the effectiveness of sound in that particular context and the ideas from it could then be generalised and used in other interfaces. However, it would be harder to evaluate and generalise to other situations. This chapter suggests adding sound in a bottom-up manner. Errors with the individual component widgets of an interface are found and sound added to fix these problems. When all the widgets in the widget set have been analysed and sound added as necessary, this sonically-enhanced widget set can be used by designers when creating new interfaces. These interfaces will then have sound built in. Different interfaces could be built using the same audio-enhanced widgets and a base of audio-enhanced applications will build up.

This chapter describes a structured, informal method of analysing the interface to find hidden information that can cause user errors. Three basic types of information are described: Event, status and mode. Any of these can cause hidden information. A method for characterising the information in terms of the feedback needed to present it is described. This is followed by applications of the analysis method to investigate various commonly used interface widgets to find out if sound can be used to improve them. For each of the widgets an event, status and mode analysis is carried out, the problems highlighted are described and the type of audio feedback needed to fix these problems suggested.

6.2 WHERE TO USE SOUND AT THE INTERFACE

There are potentially many ways that sound could be used in an interface. As Alty & McCartney showed above there is no existing set of rules that says *what* information is best presented in sound. Investigating this would have been a large problem on its own. The approach taken in this chapter is to take one type of information and investigate it in detail. The research described here proposes using sound to present information that is hidden from the user in the interface because when information is hidden errors occur. The analysis technique described identifies *where* hidden information exists and shows how sound can be used to present it. Casner & Lewis [41] suggest hidden information is a problem because (p 197):

> "Understanding how to use a computer often requires knowledge of hidden events: Things which happen as a result of users actions but which produce no immediate perceptible effect".

Considering hidden information is a similar approach to that taken by Blattner, Papp & Glinert [26] when adding sound to computerised maps. They suggested that information could become hidden because of visual clutter: Only so much information could be displayed before the underlying map was obscured. If additional information was to be displayed on a map, space must be allocated for it and eventually a saturation point will be reached where interference with the existing graphics and text cancelled out any benefit from adding more information. Blattner *et al.* suggested that sound could be used to avoid these problems and make explicit the hidden information.

6.2.1 Why is information hidden?

Information may be hidden for a number of reasons:

- ❖ *Information is not available:* It may not be available on the display due to hardware limitations such as lack of CPU power or screen size.

- ❖ *Information is difficult to access:* Information may be available but be difficult to get at. For example, to get file size and creation date information on the Macintosh a dialogue box must be called up.

- ❖ *Too much information:* Information can be hidden if it scrolls by too fast because the user's visual system is overloaded. As Blattner *et al.* [26] discussed above, presenting too much information in a small area may effectively cause some of it to be hidden.

- ❖ *Small area of visual focus:* Information can be hidden because it is outside the small area of focus of the visual system. The user may not be looking at the right place at the right time to see what is displayed.

- ❖ *Screen space:* There is a trade-off between screen space and salience of status information [149, 150]. Scott & Findlay [150] showed that by increasing the amount of screen space taken up by the status information they could decrease the time to perform an editing task. In their most salient feedback case two complete rows of the screen were taken up: One at the top the other at the bottom. This proved to be an effective way of communicating the status information but at the cost of a great deal of screen space that could otherwise have been used for user data.

- *Modes:* A mode is a state within a system in which a certain interpretation is placed on information (for example in one mode characters typed into an editor may appear on the screen, in another they may be interpreted as commands). It is often the case that the details of this interpretation are hidden from the user and mode errors result [151]. As Thimbleby ([168], p 228) says: “Typically, the user finds modes difficult because much important information is *hidden*, perhaps forgotten or not noticed by the user”. A much more detailed discussion of the problems associated with modes is given below.

How can sound help with these problems? If information is hidden because of hardware limitations such as screen size then sound could be used to overcome this. Almost all computers have the necessary sound output hardware built-in and this could be taken advantage of. Information that must be presented visually could be displayed on the screen whilst other information could be displayed in sound, saving important screen space. If the user is overloaded with visual information then some of this could be displayed in sound as the auditory sense is under-utilised and has spare capacity. One of the advantages of sound is that it is omni-directional, the user does not have to concentrate his/her visual attention on any part of the screen. Sound does not take up any screen space so displaying status information in this way leaves more space for the task the user is performing with the computer.

Dix, Finlay, Abowd & Beale [57] suggest two principles that, if satisfied, could reduce the number of errors due to hidden information: *Observability* and *predictability*. The user should be able to predict what the effects of their commands will be by observing the current state of the system. If information about the system state is hidden, and not observable, then errors will result. As Thimbleby (p 228) says: “Typically, a system gives the user no clue about how it got to its present state or does not tell the user enough for him to be certain what the current state is”. Systems hide information from the user for the reasons described above. If more information about the state of the system is made observable (or *perceivable* if other senses are to be used) then fewer errors will result. These two principles help to answer the ‘cup-of-tea’ problem suggested by Dix. When a user comes back to a computer after a break he/she may have forgotten exactly what state the system was in. If the interface provided (made perceivable) all the information needed to recover the state then the user would be able to find out where he/she had left off. Sound can be used to present information that is not displayed visually and so increase observability and predictability.

Hidden information widens Norman’s *Gulf of Evaluation* [123, 124]. Norman defines this thus ([123], p 40):

> “Evaluation requires comparing the interpretation of system state with the original goals and intentions. One problem is to determine what the system state is, a task that can be assisted by appropriate output displays by the system itself.”

Providing more information may increase the problem if it is presented in a distracting way. It may make it harder to find the important information needed to determine the system state amongst greater visual clutter. Presenting more and more information on the graphical display and making the display bigger and bigger can result in overload and the user may miss the important information. One way to overcome this is to use sound. The auditory system can provide the extra capacity needed without increasing the amount of information displayed visually.

6.2.2 ‘Action Slips’

This chapter deals with errors that result from hidden information. Information might be hidden by the system, as described above, but it might also be hidden because of errors on the part of the user. Reason [140] and Norman [124] describe the different forms human error can take. Reason describes three types:

- ❖ *Mistakes:* The user has an incorrect plan of action but carries it out correctly leading to an incorrect result.

- ❖ *Action slips:* The user has a correct plan of action but fails to carry it out correctly.

- ❖ *Rule-based mistakes:* These are a mixture of mistakes and action slips. Reason says they arise from the application of inappropriate diagnostic rules.

Many of the usability errors with interface widgets that this chapter deals with occur because of action slips. The user has a correct plan of action but fails to use the widget correctly and an error results. Reason ([140], p 8) suggests:

> “Two conditions appear to be necessary for the occurrence of these slips of action: The performance of some largely automatic task in familiar surroundings and a marked degree of attentional capture by something other than the job in hand”.

This type of error commonly occurs with expert users. They perform many interface tasks automatically without monitoring the feedback from the interaction because the task is well known. Dix *et al.* [57] describe a problem with users slipping off graphical screen buttons, which is an error of this type. The use of graphical buttons is a commonplace activity for users of graphical interfaces. Buttons are very common and occur in many types of interactions. Much of their use will be automatic and only given low-level attentional control. As mentioned, a necessary condition for the occurrence of a slip is the presence of attentional capture associated with either distraction or preoccupation. In this case the capture would come from preoccupation with the main task being undertaken. This may be, for example, the preparation of a paper and any graphical button presses are a small, subsidiary part of this. The writer’s attention will be focused on the paper being written, not on the graphical button press.

Once an error has occurred it must be detected. Reason (p 157) says:

> “Making a postslip attentional check does not of itself ensure the detection of the error. Detection must also depend upon the availability of cues signalling the departure of action from current intention”.

Visual cues may (or may not) be given to indicate that the operation the button invoked did not take place. However, these may not be noticed. The user will have reached *closure* (see the section on naïve psychology below for more on this) on his/her current task and gone on to the next one. He/she may no longer be looking at the location of the last interaction, where the error is being signalled. Thus, the error may not be noticed until it is too late to correct it (for example, with only a one-step undo facility). The information is hidden in this example because the user is not looking at the right place on the screen at the right time to see the error feedback. The different capabilities of the auditory system can provide advantages here. The addition of more demanding audio cues may help capture the user’s attention and indicate that there has been a problem with the button press.

The discussion in this section has demonstrated some of the problems due to hidden information in the human-computer interface. To overcome them this thesis suggests using sound. A method is therefore needed to find out where hidden information exists so that it can be made explicit.

6.3 THE EVENT, STATUS AND MODE (ESM) ANALYSIS TECHNIQUE

A method is needed for finding where the hidden information exists in an interface. This can be done by modelling interactions in terms of event, status and mode information. In the following section, a description will be given of the event, status and mode (ESM) analysis technique, what it is based upon and why modes were introduced into it.

6.3.1 Dix’s event and status analysis

Analysing interactions in terms of event and status information was first put forward by Dix and colleagues [57, 59, 60]. A brief description of his analysis method follows. Dix *et al.* ([57], p 325) describe it as an ‘engineering’ level technique. They say: “An engineering approach is built upon theoretical principles, but does not require a deep theoretical background on the part of the designer”. Dix’s technique is built on formal models of interaction but the interface designer using it does not need to know these in order to employ it effectively. It is also built on what Dix calls 'naïve' psychological knowledge, which is used to predict how particular interface features affect the user (see below). These foundations make the technique powerful and easy to use.

One of the advantages Dix claims for this technique is that it can be used by designers who have no knowledge of the underlying formal models of interaction. It is also easy to apply because it uses common concepts that can be applied at different levels of interaction. It is for these reasons that this technique was chosen as the method for finding where to use sound at the interface. This means, however, that it does not have some of the advantages of a formal method for interface analysis (such as Z or CSP [95, 159]). These are more abstract and allow a designer to concentrate on what the system does rather than how it does it. The event, status and mode analysis is not as precise as a formal notation and does not allow the designer to formally reason about the properties of the analysis to find inconsistencies.

Dix claims the strength of the method is that a single framework can be applied at many different levels in an interaction, ranging from the application, through the interface to the user's perception. The way the analysis technique works is that it considers the different layers within the system, such as the user, screen, dialogue and application. It looks for events and status changes at each of these levels. These, combined with the naïve psychological analysis of the presentation/user boundary, allow the designer to predict failures and suggest improvements. This was another important reason for using the technique. It was necessary to find problems at the interface so that they could be corrected using sound. Dix's technique would provide this and suggest improvements to the interactions investigated.

Naïve psychology

Dix says that, in order to predict the effect of interface techniques, we need to use some naïve psychological knowledge. This will indicate where the user's attention is likely to be focused and what kinds of feedback will be salient. Dix gives several examples. He says it is sometimes possible to predict where the user is looking:

❖ *Mouse:* When the user is positioning the mouse over a target he/she is likely to be looking at that location because of the hand-eye coordination required. The attention may not stay long after the target has been hit.

❖ *Text insertion point:* While typing the user will intermittently look at the text typed and hence the current insertion point. However, because touch typists may look at their source document or hunt-and-peck typists at the keyboard, it is less likely that users will be looking at the insertion point than at the mouse.

The only time attention may be focused on the insertion point is when it is being moved over large distances using the cursor keys.

- ❖ *Screen:* It is reasonably safe to assume that the user will look at the screen intermittently. However, there is no guarantee that any particular message or icon on the screen will be noticed, only that very large messages spread across a large part of the screen will probably be noticed.

If it is known where the user is looking then events can be signalled noticeably at that point, for example by changing the cursor. If it is not known where he/she is looking then other methods must be used. Sound is very useful in this case as it can be heard from all around and is not restricted to a small area of visual focus. If the user is typing, moving the mouse or pressing on-screen buttons then it is likely that he/she is using the machine, even if it is not certain exactly where visual attention is focused. Sound is very useful in this case as it allows information to be presented to the user even if it is not known where he/she is looking. Dix also notes that peripheral vision is good at detecting movement and this could be used for capturing attention.

The final point Dix discusses is *closure*. When a person completes a task he/she experiences closure and goes on to the next task. One classic example of this is with Automatic Teller Machines (ATMs) [58]. The user’s task is to withdraw money. When the money has been withdrawn closure will be achieved and he/she will go on to do something else (such as spending the money). If the ATM gave the money before returning the bank card then the user is likely to walk away and leave the card because he/she achieves closure when the money was dispensed. In terms of the interface, this may mean the user moves off a target as soon as he/she thinks it has been hit, when in fact it may not have been. An example of this type of problem with on-screen buttons is described in greater detail later in the chapter.

This section has given an overview of Dix’s analysis technique. It has shown that the technique is easy to apply because it uses simple psychological assumptions and a designer does not need to know about the formal models it is based on. It does have the disadvantage that it is not as precise as a formal modelling technique. The following sections describe in detail the three types of information in the analysis technique. First, events will be described.

6.3.2 Events

An event marks something that happens at a discrete point in time. Events are caused by actions on the part of the user (mouse clicks, button presses) or the system (mail arriving). There can be input events (mouse clicks) or output events (a beep indicating an error). The same actions may cause different events under different circumstances.

For example, the user clicking the mouse in one window may select an icon but in another may position a cursor.

Dix suggests that there are two time points to an event: The event which occurs and the perception of its occurring. As Dix ([60], p 8) says: "…there may be a lag between the *actual* event and the *perceived* event". If an event occurs and changes the status (for example, new mail arrives putting a message in the mail window) the user might not notice straight away. The actual event is the mail arrival but the user might not perceive it instantly. It is only when he/she looks at the mail window that the new mail will be noticed (the perceived event). In some cases there may be no perceived event, so that actual event is missed by the user. The perceived event may be missed, for example, because the user was not looking at the right place on the screen at the right time. An actual event that does not permanently change the status (for example a screen flash to indicate an error) will be never be perceived by the user if he/she is not looking at the screen at the time because it leaves no continuing status information. There may also be a perceived event with no corresponding actual event: The user may have thought he/she heard a beep from his/her computer but it actually was from someone else's.

If the event is not perceived by the user then it is hidden and this can result in errors. There are two reasons why the event might not be perceived. It might be that the event is not *perceivable* - there is no way that the user can see or hear it because the system does not display it (for example, the information is not displayed due to hardware limitations). The alternative is that it might not be *perceived* - the user could perceive it but did not (perhaps he/she was looking at the wrong part of the screen). The aim of the research described here is to make these two types of hidden events explicit.

6.3.3 Status

Dix describes status information as any phenomenon with a persistent value, for example the location of a mouse, an indicator, a temperature gauge or CPU load monitor. Much of the display in a graphical interface is status information. Status information is what the user can perceive about the internal state of the system. The state of the system is the complete set of values of the internal variables and components of the system. If there is information about the state that is not displayed as status information then it will be hidden from the user. If important state information is not displayed in the status (i.e. is hidden information) then errors can occur.

Events may change the status information. An event such as a mouse click in a scrollbar may change the status information displayed in a window because it scrolls. Another example is clicking the 'OK' button in a dialogue box. This causes the box to disappear, changing the status of information displayed on the screen. If an event changes the state of the system then this should be reflected in the status, if it is not then the hidden information could cause errors.

The status information can be displayed, or *rendered*, in many different ways, for example graphically, textually, sonically or by a combination of these. Earlier in this chapter the problems of presenting status information graphically were discussed. Scott & Findlay [150] showed that, if it was rendered graphically, much screen space had to be devoted to make it noticeable. This thesis suggests that rendering status information in sound is possible and could overcome some of the problems associated with visual representations. There are similar problems with status as there are with actual and perceived events. For example, a user might not notice the grey border around a window indicating that it is the active one.

Dix discusses another property of events and status: *Polling*. A status can be turned into an event by looking at it to see whether a particular value has been reached. For example, a person may look at their watch every few minutes to see if the time has reached 12:00. The watch normally gives status information. When the time reaches 12:00 then it is perceived as an event. Events can also turn into status information. In a process control system the value of the variables, such as pumps or valves, will only be monitored intermittently and the value of these then updated on the screen. If the update is often enough, the user will perceive continuous status feedback on the screen. These examples show that events and status can change.

When using this analysis technique the differences between events and status are used to classify interactions. Dix has used it in two main examples [57]: Email and screen buttons. In the first example he investigated an email system and looked at the types of feedback that could be used to make the actual event of mail arrival a perceived event for the user. In the screen buttons example the technique showed a problem in the feedback from the button. This meant that it was easy for the user to slip off the button by mistake and not notice (see section 6.6.3 for a detailed discussion of this).

This section has given background to the event and status analysis technique as described by Dix. It is based on a formal model of the interface but knowledge of this is not required by a designer in order to use it. This does however mean that it is not as precise as a formal analysis method. Dix suggests his technique can predict failures and suggest improvements to interactions. It was decided that it should be used to look at the places where sound might be used at the interface. Sounds can be of two types: Discrete event sounds (beeps) or continuous status sounds (air-conditioning sounds, fans). In interfaces that use sound, most use event sounds, for example beeps to indicate errors, tapping sounds to indicate selection of items. Few use status sounds by design, often these occur by luck, for example, cooling fan or hard disk sounds. One of the aims of this research is to show that status information can be rendered effectively in sound. Using this analysis technique would allow the identification of such information and therefore permit it to be rendered in sound at the interface.

Event and status information have now been described. The next section discusses modes: What they are, why they are a problem and previous work to investigate them. It then discusses why they should be added to the event and status analysis.

6.3.4 Modes

It was decided as part of this research that modes should be added to extend Dix's event and status analysis. Modes are very closely linked to events and status (as will be shown below) and they cause some of the main problems of hidden information at the interface. Modes at the human-computer interface have been investigated ever since Tessler complained about being 'moded-in' in 1981 [167]. However, as Sellen, Kurtenbach & Buxton report [151], even now there is little relevant literature in the area. Two of the main experiments are described below.

What are modes?

A mode is a state within a system in which a certain interpretation is placed on information. For example, in one mode characters typed into an editor may appear on the screen, in another mode the characters may be interpreted as commands. It is often the case that the details of this interpretation are hidden from the user (or perhaps forgotten due to overload of short-term memory, or not noticed because the user was not looking at the required part of the screen). Mode ambiguity occurs when the status does not provide enough information to indicate which mode the system is in. The definition of modes can be extended to cover a set of states. Modes group a set of event and status information. Frohlich (reported in [3], p 38) confirms this, defining modes as: "States across which different user actions can have the same effect". In one mode a set of user actions is possible and this causes a particular set of events. A set of status outputs describe the state of the system. In another mode the same set of user actions may cause different events and there will be different status feedback. Indeed, a different mode might have a completely different set of user actions. For example, in a wordprocessor key-strokes will cause characters to be displayed on the screen. The window and the text it displays provide status information about being in word-processor mode. If the user switches to a drawing package, the set of legal user actions might change to mouse clicks and drags. Status information would change to reflect the new mode. There would be the graphics displayed in the window, a tool palette and the window itself might be in a different place on the screen. Therefore, a mode groups a set of events and status information.

Johnson [97] and Johnson & Englebeck [98] say that there is no generally agreed definition of mode. The definition given above is useful for the purposes of this thesis but is not the only one. One of the most general Johnson describes thus (p 424): "…a system has modes (i.e. is moded) if the effect of a given user-action is not always the same". This is a very general description which can mean that almost anything could be described as a mode. Moving the mouse from the background to an icon could be describe as a mode change because pressing the mouse button would have a different effect in each case. He gives examples to show that modes do not only occur in computer systems but in other machines such as cameras, stereos and ovens. His definition of modes fits well with the event and status analysis as user actions cause events. These events have different effects depending on the mode.

A further definition of mode errors is reported by Sellen *et al.* ([151], p 142): "Mode errors … occur when a situation is misclassified resulting in actions which are appropriate for the analysis of the situation but inappropriate for the true situation". The user observes the state of the interface, through the status information displayed, but misclassifies it and the predicted effect of the commands is different from reality. So, situations can be misclassified because there is not enough status information informing the user of the state of the system. It can be seen that mode and status information are strongly linked. Sellen *et al.* go on to say (p 143):

> "It is not clear that we can ever hope to completely eliminate the problems associated with modes, but it certainly seems possible to reduce them. One obvious solution seems to be to give users more salient feedback on system state".

If the user was given more status feedback then he/she would be less likely to misclassify a situation and the effects of mode ambiguity would be reduced. Thimbleby [168] describes many modes as *spatial modes:* The meaning of the mouse button depends on the position of the mouse. The position on the display where the current dialogue is taking place is all-important. If the user knocks the mouse into a different window, gets distracted or comes back to the machine after a break, then he/she may try to communicate with one part of the system, but the system thinks the user is communicating in a different part - mode ambiguity. Often users do not give the system their undivided attention, they may be doing other things and miss the event which indicated the mode change. They may not remember the exact status of the display and so may not notice that a change in status has occurred.

Modes are not always bad. Thimbleby says that the restricting nature of modes can be good for users in some circumstances as it can stop them from doing things they do not want to do. If, for example, the user could only delete all the files in his/her directory in one mode and not any others, then it is unlikely to be done by mistake. The modes must, however, be made explicit so that users know when they are in 'delete file' mode.

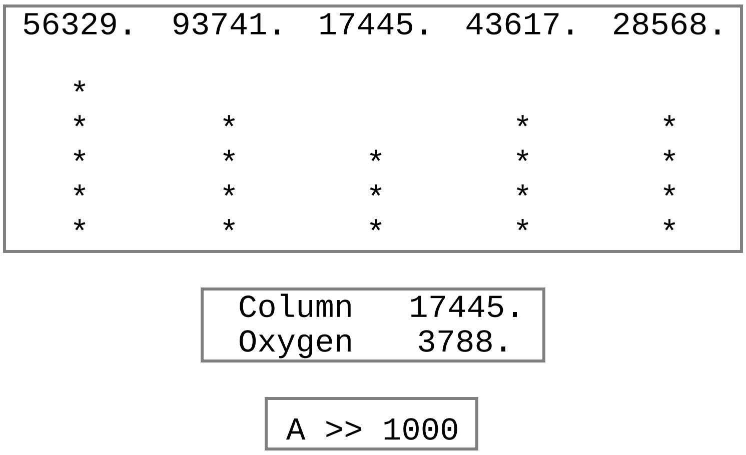
The consequences of most mode errors at the human-computer interface are often only minor inconveniences which are usually easily reversible. However, errors in more complex systems, such as aircraft or nuclear power plants, can have much more serious outcomes so their prevention is important. Errors are not the only problem that can occur due to mode ambiguity. As Sellen *et al.* suggest (p 143):

> "In some cases, the user may diagnose the correct mode, but only after experiencing confusion or uncertainty. In such cases, the appropriate measure is in terms of the cognitive effort or decision time required to deduce the system state".

These types of problems are important to consider as solving them would speed up user operation of systems and reduce frustration. Now that modes and the problems they cause have been discussed, two investigations into how to deal with them will be described.

Two investigations into the problems of mode

Modes have been known to be a problem for a long time, but as Sellen *et al.* say little work has been done to investigate ways of improving them. Monk [117] performed some early research into modes and mode errors. Interestingly, as will be shown below, he chose to use sound to make the modes explicit.



```
56329.  93741.  17445.  43617.  28568.

   *
   *       *               *       *
   *       *       *       *       *
   *       *       *       *       *
   *       *       *       *       *

           Column   17445.
           Oxygen    3788.

              A >> 1000
```

Figure 6.1*: The screen used in Monk's experiment (from* [117]*).*

Monk wanted to find out if providing sound cues to indicate mode would alert users to potential errors. He says (p 318):

> "The normal way to signal that a system is in a particular mode is by means of the display. A particular prompt or cursor may be used, a message may be displayed or the screen layout may change. The problem is that much of the time the user will not be looking at the display. A touch typist using a word processor may be looking at the source document, a poorer typist may have to look at the keyboard. Users of other systems may be similarly distracted for example, while they are interacting with a client or customer".

The system he used was a computer game where the subjects had to control a chemical plant. The display is shown in Figure 6.1. The five columns of asterisks showed the quantity of oxygen in each of five reactors, if no action was taken by the user the quantity of oxygen decreased. The user's task was to keep them topped up. If any reactor ran out then the game was lost. Subjects had to keep the columns topped up for a fixed amount of time to win. To top up a column the subjects had to type in the five-digit column identifier when in column identifier mode and then change to adding mode and specify the quantity of oxygen (the 'A' prompt in Figure 6.1 above). Time pressure was applied to induce more errors. Two groups were used in the experiment. Both had keying contingent sound but only the experimental group heard different sounds depending on mode; the control group heard the same sound in both modes. The experimental group heard a high pitched tone (2200Hz) when in column identifier mode and a low pitched tone (250Hz) when in adding mode.

The errors Monk tested for were: Typing an amount of oxygen when in column identifier mode and typing a column identifier when in adding mode. The results of his experiment showed that the experimental group made one third less mode errors than the control group. They did not make errors and then recover from them more quickly, they made fewer mode errors. Monk suggests that this is due to a better general awareness of the modes in the system amongst the members of the experimental group. Monk says (p 323): "…on the basis of the results reported here, mode-dependent keying-contingent sound seems to be a promising approach to the problem of mode-ambiguity". This is a very important result; it showed that if the general awareness of modes can be raised users will make fewer mode errors.

Sellen, Kurtenbach & Buxton [151, 152] investigated methods of indicating mode in the *VI* text editor. In *VI* there are two modes: Edit mode, where characters typed appear as text on the screen, and Command mode, where characters typed are interpreted as commands. They used two methods to indicate the mode: Kinesthetic feedback via a foot-pedal and extra visual feedback. Sellen *et al.* [151] (p 145) say: "…we maximised the saliency of the visual feedback by changing the entire screen color [sic] to a dark pink color while in edit mode". In command mode the screen was white. In the kinesthetic condition pressing and holding the foot-pedal down put the system into insert mode, releasing the pedal put the system into command mode. Their hypothesis was that, because the user actively had to generate and maintain the mode when using the foot-pedal, it would provide the more effective feedback leading to fewer mode errors. Subjects had to edit a document to insert a given string after all the capital letters.

Their results showed that using the foot-pedal was beneficial. The pedal caused a greater reduction in mode errors than the visual feedback. It also led to significantly faster times than when just the keyboard was used. Sellen *et al.* suggest that the foot-pedal was effective because it reduced the cognitive load of the task. They give several reasons for this:

- ❖ *User-maintained:* The kinesthetic feedback had to be actively maintained by the user. In the visual condition users received the mode information passively.
- ❖ *Sensory channel:* The visual channel is inherently more avoidable than the kinesthetic one - users could choose not to look at the display but they had actively to press the pedal.
- ❖ *Visual attention:* The visual feedback may have competed with the visual nature of the editing task. Searching the screen and monitoring the outcome of keystrokes might have meant that fewer attentional resources were available to monitor the screen colour.
- ❖ *Distribution of tasks:* The mode was sustained by the pressure of the foot on the pedal and the task involved the fingers in the kinesthetic condition. In the visual condition mode changes were accomplished via the fingers also. The separation in the former case might have proved beneficial.

How does presenting mode information in sound fit with their analysis? Sound would not be user maintained, it would be passive like the visual feedback. Sound, however, would be more demanding than graphical output so in this way has some of the advantages of the kinesthetic feedback. Sound would not compete with the visual nature of most computer tasks and so have the benefits of the foot-pedal. Sound is also a different sensory modality so the advantages of distribution of tasks across senses would be gained.

This experiment shows the advantages that can be gained from displaying mode information to the user. Fewer mode errors will result and switching between tasks could be speeded up. Sellen *et al.* suggest two main design principles to come from their work. The first is that the modality of the feedback is important; as the visual channel becomes overloaded, other senses must be used. The second is that care should be taken so that feedback does not interfere with the primary task. This thesis suggests using sound to overcome these two problems.

It is interesting to note from the two examples of investigations of modes described above that neither of them solved the problems by using extra graphical feedback; both used alternative sensory modalities. The same approach is taken in this thesis.

6.3.5 Bringing events, status and mode together

It was decided that modes should be specifically brought into the event and status analysis so that they could be examined explicitly. As the above examples have shown, modes can cause problems in the interface because often there is not enough status information to make them explicit. As the reason for using this analysis was to find hidden information in the interface modes had to be included.

As discussed at the end of the section on status information, analysing an interface in terms of event and status fits well with the potential use of sound at the interface. Sound can be discrete or continuous in a similar way to event and status information. Including mode in the analysis does not change this. The event, status and mode analysis will model situations where information is hidden. Dix has shown that his technique is powerful when considering just the two types of information and extending it to use mode should make it even more effective. The technique was chosen to be used as part of this research because the way it models information matches the way sound can be produced. It does not require detailed knowledge of formal system modelling to use and it can identify hidden information which has been shown to be a problem at the interface.

6.4 CHARACTERISATION OF THE FEEDBACK

Dix's analysis method finishes when the event and status information has been identified. It shows where errors occur because of incorrect event and status feedback and can suggest improvements. It does not suggest what feedback is needed to correct the problems found. If his technique is to be used as part of the structured method for integrating sound into human-computer interfaces then it must connect to the work on earcons described in the previous chapters. Dix's work is extended here so that when the event, status and mode information has been extracted it is characterised in terms of the feedback needed to present it to the user. The earcon guidelines can be used with this characterisation to design the sounds necessary. The characterisation described here is based around that of Sellen, Kurtenbach & Buxton [151, 152]. They use five dimensions of feedback: Modality of delivery, user versus system maintained, action-dependent versus independent, transient versus sustained and demanding versus avoidable.

The first dimension deals with the sensory modality through which the feedback is delivered. As discussed above, this thesis suggests that any new information should be rendered in sound to avoid increasing the visual complexity of the display. Sellen *et al.* showed that other sensory modalities could be used but the work in this thesis investigates the addition of sound to overcome the problems so this dimension is not appropriate here. The next dimension is user or system maintained feedback. The experiments that Sellen *et al*. conducted (described above) showed that kinesthetic user-maintained feedback was more effective than visual system-maintained feedback. As no new physical methods of controlling feedback were to be introduced as part of this research this dimension was not used. The following sections describe the other three of Sellen's dimensions plus a fourth, added as part of this research. As well as describing the dimensions, they are discussed in terms of the event, status and mode analysis.

6.4.1 Action-dependent versus action-independent

Does the feedback depend on a user or system action taking place? Events are action dependent; an action on the part of the user or the system must occur for an event to take place. For example, the user clicks the mouse button and this causes an event such as selecting an icon. Feedback is only given when the event occurs. Status feedback, however, is action-independent: It is not affected by user/system actions. Status feedback continues whether there are actions or not. Status may be changed by events but it continues independently of them. For example, an event may cause a dialogue box to be displayed. The status feedback from this will continue until the user presses the OK button regardless of whether the user moves the mouse or presses keys on the keyboard. Modes, like status, are initiated by events and continue until a further event changes the system to a different mode. For sound feedback, action-dependent delivery would mean sound occurred when some action took place, for example Monk's keying-contingent sound [117]. Action-independent delivery would mean that feedback was given without an action taking place, for example a constant tone indicating what mode the system was in.

6.4.2 Transient versus sustained

Is the feedback sustained throughout a particular mode? Events are transient, they occur at momentary, discrete points. Transient delivery is therefore useful for presenting event information. For example, a short beep to indicate an error. Events are atomic: Nothing else can take place during an event (this depends on the level of abstraction, as will be discussed below). Status information continues over time so sustained presentation is needed. For example, a window on the screen displaying the contents of a document is sustained. Sustained sounds can be habituated by listeners [39]. The user does not actively have to listen to the sound. The sound will be perceived again only when it changes in some way (when an event occurs) or if the user consciously chooses to listen to it. Status information is non-atomic: Actions and events can take place whilst the status information is presented. Modes may be sustained or transient. A mode might last for a long time (like the window) and be sustained or for a short time (a screen button press) and be transient. Modes may be atomic or non-atomic.

6.4.3 Demanding versus avoidable

Can the user avoid perceiving the feedback? Events should be demanding as they mark important occurrences in the system. The classic example of this in the interface is a demanding beep to indicate an error event. The user needs to know that an error has occurred. Status information should be avoidable. It exists over the time and the user should be able to choose to sample it only if he/she wants to. For example, the data in a window on the screen should be avoidable. The user can look at the window if required but should not be forced to see it. This is not always the case as the user may not be interested in some events (for example, the arrival of some types of junk email) and may not want to miss some types of status information (for example, an alarm). In terms of Dix's analysis, with a demanding event there will only be a short time between the actual event and the perceived event and the user will always perceive the event. Avoidable feedback may have a longer time between the actual event and the perceived event. In fact, listeners may not notice the feedback at all and never perceive the event. Modes should be demanding. Often feedback from a mode is avoidable (when it is there at all), the user does not observe that he/she is in the mode and mode errors can then occur.

This aspect of the categorisation can capture the urgency of the information to be presented. The work by Edworthy *et al.* [64, 65] (described in Chapter 3) can be used to make sounds demanding (more urgent) or avoidable (less urgent) as required. As discussed above and by Scott *et al.* [149, 150], presenting information visually so that it is demanding can take up much screen space. Presenting demanding information in sound is easier to do as sound is, by its nature, attention grabbing. This means, however, care must be taken when creating avoidable sounds so that they are not demanding by mistake.

6.4.4 Static versus dynamic

Does the feedback change whilst it is presented or is it constant? This extra dimension of feedback was added as part of this research because it was not captured by Sellen *et al.*'s classification. Events are static; they only occur for a moment of time and indicate that one particular thing has happened. Status information can be static, for example a window onto a file directory, or dynamic and change over time, for example a CPU load indicator. *Animated icons* [12] are an example of dynamic feedback. These change to capture the user's attention or to give information about their state over time. One other example is *metawidgets* [84]. In this system, widgets may move to the edges of screen of their own accord if they are not used for a period of time. Modes are static; they do not change their meaning whilst they are presented. Sound can be static or dynamic, for example a constant tone is static and music is dynamic.

If many events occur together then they may be seen as dynamic status information when looked at on a larger scale. At a lower level of granularity, status could be viewed as being composed of many events, each change in status is an event. A change in status will only be perceived as an event if it is salient to the user.

The feedback from each of these categories is not necessarily independent. For example, dynamic visual feedback is more demanding than static feedback because the user's eye is drawn to the changing stimulus (see the section on naïve psychology above). In the same way demanding audio feedback captures attention. In order to create demanding feedback a high volume, static sound could be used or, alternatively, a lower volume, dynamic sound. In the latter case, it would be less annoying for nearby users but just as demanding for the primary user. Therefore, a demanding sound could be created that was both attention-grabbing for the primary user but not annoying for other users at the same time.

This categorisation converts the raw hidden information from the event, status and mode analysis into a structured form which can then be converted into sound. Once the categorisation has been used the designer will be able to create sounds to represent the hidden information. The previous sections have described the event, status and mode analysis method. It could be used by an interface designer, who does not have knowledge of the formal models it is based on, to find potential areas of error in his/her interfaces. The results from the analysis could be used to improve the graphical feedback from interactions to overcome the problems of hidden information. However, this thesis suggests that this would clutter the interface further and overload the visual system.

6.5 PREVIOUS WORK ON MODES RE-EXAMINED

Monk and Sellen *et al.* analysed problems caused by modes. How did their solutions fit into the analysis technique that has just been described? Monk added feedback that was action-dependent (the user had to type a key to hear the feedback on which mode the system was in), demanding (the sound captured the user's attention - they could not avoid listening to it), transient (the sound was a short beep) and static (the sound for the mode did not change). There were two modes and each had a different sound. This feedback made the hidden mode explicit by adding sound to keystroke events.

Sellen *et al.* wanted to make the hidden modes explicit in the *VI* text editor. They tried to do this in two ways. First, they made the screen pink to indicate edit mode. This feedback was action-independent (the screen stayed pink until a change mode event occurred), avoidable (the users did not see the feedback), sustained (the feedback continued while the mode was in operation) and static (the feedback did not change).

One might have thought that the pink screen would be demanding but, due to the overloading of the visual system (the user had to perform a visual editing task), the feedback was avoidable. The other method they chose was to make the user actively maintain the mode (action-dependent) by holding down a footpedal. This made the feedback demanding and so users did not miss it.

6.6 EVENT, STATUS AND MODE ANALYSIS APPLIED

This informal analysis technique can be used to investigate known problems with interactions and find what might be causing the mistakes. It can also be used to look at new interactions to find where there are likely to be faults. The technique has been used to analyse many widgets. These analyses will be applicable to many different interfaces because, as Myers' [120] shows, most widgets vary little in their general design across different interfaces.

To use the event, status and mode (ESM) analysis technique first think of the 'generic' or perfect widget, for example a button, which provides all the information required and where nothing is hidden. This can be done by creating scenarios depicting the interaction with buttons and mapping out the possible types of events, status and modes that could occur. Identify all the event, status and mode information in the interaction and the feedback required to present it using the descriptions given previously. Then do the same for the real button, identifying the information actually present. If there is less in the real button than the ideal, generic one then that is where hidden information exists. If there is more, then there may be redundant information that could possibly be removed. This is similar to the approach taken by Kishi in the SimUI system [101] where a 'model' or expert user's interactions were compared to those of normal users and where the differences occurred were usability problems. One alternative approach to using the method is to first think of the real widget and analyse it in terms of events, status and modes and then categorise the feedback. Then, using this information, construct a generic widget which deals with the problems of the real one and makes explicit the hidden information.

The correct level of abstraction should be used when finding the event, status and mode information. Dix *et al.* call this a problem of *granularity*. As an example consider dragging an icon. If a high level of abstraction is chosen dragging is an event that causes an icon to move (change status). At a lower level, dragging is a 'press the mouse button over the target' event, a 'move' event (that changes the status) and a 'release mouse button' event. At a lower level still, dragging is made up of many small move pixel events. The level of abstraction of the analysis should be chosen depending on what is being investigated. If there is a problem with the user not releasing the mouse button correctly over a target then looking at the interaction in terms of dragging as a single event is no good because too much detail is hidden; the abstraction level is too high. Considering dragging in terms of pixel changes would be too low because the analyst would not be able to see the important changes in amongst the other information. In this case, using the middle level abstraction described above would provide the correct amount of information. So, the correct level is one where all the information necessary to investigate the problem is available but not lost amongst too much noise.

The rest of this chapter shows the application of the ESM technique to various interface widgets to find problems. A simple example analysis of a caps-lock key is given first to show how the method is used and then five other widgets are analysed: Dialogue boxes, buttons, menus, scrollbars and windows. Each section begins with a description of the problem in the real interaction and then shows the ESM analysis and results.

6.6.1 Caps-lock key

Generic caps-lock key:	**Feedback:**
Mode Two modes: Upper-case mode and lower-case mode. When caps-lock key engaged system is in upper-case mode and vice versa. Only really need to indicate unusual condition - upper-case mode	**Mode** Action-independent, demanding, sustained, static
Event Caps-lock off ➜ caps-lock on Caps-lock on ➜ caps-lock off Key presses	**Event** Action-dependent, demanding, transient, static for all events
Status 1. Caps-lock key 2. Case of letters on screen	**Status** 1. Action-independent, demanding, sustained, static 2. Action-independent, avoidable, sustained, static
Real caps-lock key:	**Feedback:**
Mode Two modes: Upper-case mode and lower-case mode. When caps-lock key engaged system is in upper-case mode and vice versa. Only really need to indicate unusual condition - upper-case mode	**Mode** Action-independent, **avoidable**, sustained, static
Event Caps-lock off ➜ caps-lock on Caps-lock on ➜ caps-lock off Key presses	**Event** Action-dependent, **avoidable**, transient, static for all events
Status 3. Caps-lock key 4. Case of letters on screen	**Status** 3. Action-independent, **avoidable**, sustained, static 4. Action-independent, avoidable, sustained, static

***Table 6.1**: ESM analysis of a caps-lock key.*

As an example of how to use the technique, an ESM analysis of a caps-lock key will be performed. The problem with the key is that it can be pressed by mistake so that text is typed in upper-case. This might not be noticed by ‘hunt-and-peck’ typists because they are looking at the keyboard or copy-typists as they are looking at the source document. The user then has to go back, delete the text and re-type it in the correct form. Alternatively, the user may forget to turn caps-lock off and type extra characters in the wrong case again because he/she was not looking at the screen.

When performing the analysis it is often useful to draw up a table of the information extracted. Table 6.1 illustrates the ESM analysis for the caps-lock key. In the table, items in the real part of the analysis that are different from the generic, ideal ones are emboldened. Items not emboldened are the same. To use the technique, first identify all the modes present in the interaction. Then look for the events and then the status information. The combination of the different types of status information should match that required by the mode. In this case, the three types of information must be identified by looking at the caps-lock key and considering the feedback present with the descriptions of events, status and modes supplied above; then see which parts of the feedback fit into which category of information.

Generic caps-lock key

There are two modes: Upper-case mode and lower-case mode. Lower-case is the default setting so really only upper-case mode, the unusual setting, needs to be signalled. The mode will be action-independent (it will continue until one of the events occurs, typing, for example, will not affect it), demanding (the user should know he/she is in upper-case mode), sustained (the mode will continue until the caps-lock key is pressed again) and static (status feedback on the mode will not change until a mode change event occurs). In order to make the mode explicit there should be status information matching this analysis.

The mode has three events: Turning caps-lock on, turning it off and key presses. For both of the caps-lock key events the feedback is action-dependent (the user must press the key), demanding (the user should know that the key has been pressed), transient (the key is pressed and the user moves on to their next task) and static (the feedback about the button press does not change). Key press events are the same as the caps-lock events. The key press events change the status of the screen feedback causing a letter to be displayed. The actual events will be perceived as soon as the key is pressed because the user actively has to perform the actions involved.

There are two types of status information to make the mode explicit: Feedback about the caps-lock key and screen feedback. The feedback from the key is action-independent (it stays pressed until the caps-lock off event occurs), demanding (the user should know if the key is pressed or not), sustained (the feedback continues until the key is disengaged) and static (the feedback from the key stays the same until it is pressed again). Feedback is also received from the screen: The user can see if the letters typed are in upper or lower case. The feedback is action-independent (the screen feedback continues regardless of user action, but will be changed by keystroke events), avoidable (the user can choose not to see the feedback by not looking at the screen), sustained (the upper or lower case letters stay on the screen) and static (the letters do not change after they have been typed). Screen status feedback is changed by key press events: Characters are displayed on the screen. The feedback is action-independent because status feedback is only changed by these events. The combination of both types of status information makes the hidden information in the mode explicit.

Real caps-lock key

Now the real caps-lock key will be analysed. The modes are the same as before: Caps lock on or off. This time the mode is avoidable (resulting in the errors described above). The events in the mode are the same as before but feedback about pressing the caps-lock key is avoidable. It can be pressed by mistake, so that there is no perceived event and the user does not notice the mode change. Key press events are avoidable because the user might press keys by mistake and not notice.

The generic caps-lock key has two types of status feedback: One demanding, the other avoidable. This is combined with demanding event feedback so that the user knows when the caps-lock key has been pressed. These provide enough feedback to make the mode explicit. In the real situation, however, feedback on the state of the key is avoidable: The user might look at the physical key on the keyboard but they might not. Therefore, both types of status feedback are avoidable and the mode is not made explicit. In order to make the state of the key more demanding lights are often used, either on the caps-lock key or at the top of the keyboard. These indicate the mode but are often outside the area of visual focus and so are still avoidable.

To overcome the problems and make the mode explicit, demanding audio feedback could be provided for the caps-lock events. When the key was pressed a tone could sound. This would enable the user to correct the error. This would solve the first problem described above.

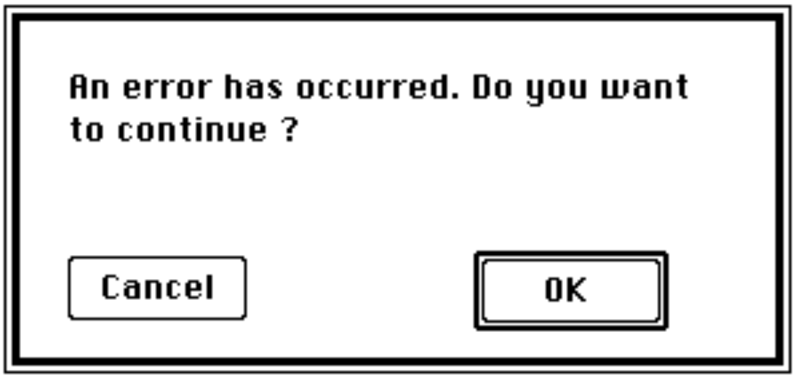
The second problem (forgetting that the caps-lock key is engaged) would not be solved by the solution above because it does not display the current state of the button as status information. There are two other possible solutions. The first would be to add an action-independent, demanding status sound (see 3 in the real caps-lock key in Table 6.1). This sound would continue for as long as the mode existed. This would cure the problem because the user would be able to tell the system mode. This does, however, have one drawback: A demanding, sustained sound is likely to become annoying very quickly. The caps-lock key is not an important enough problem for this type of feedback to be necessary. In order to get around this problem the key press event feedback of could be changed. The action-dependent, avoidable feedback could be made demanding by adding sound. This would then work like Monk's keying-contingent sound (described above). Combining the feedback for the caps-lock key and key press feedback an interaction would progress as follows: A key-click sound would be played when the caps-lock key was pressed down or released. When users typed and the caps-lock key was down, beeps would accompany each key press. This feedback would indicate that the system was in upper case mode. This feedback would be less annoying and, as Monk showed, effective at reducing mode errors. This shows some of the different properties of sound and graphics. Continuous visual feedback would not be annoying for the user whereas continuous sound feedback might be, unless it was carefully designed.

In many ways, this is a similar problem to that investigated by Sellen *et al.* [151] (described above). If only a shift key was available, i.e. one that did not latch and stay down, the mode would be demanding due to the kinesthetic feedback. The user would actively keep the mode in operation, if he/she took a finger off the shift key the system would revert to lower-case mode.

Now that this simple example of the application of the ESM analysis technique has been described, more detailed analyses will be undertaken. Some of the most important and common widgets will be examined to find out what problems are affecting usability.

6.6.2 Dialogue boxes

Dialogue boxes have two main uses [58]. They can be used by the system to alert the user to errors or other important information, for example Figure 6.2. Alternatively, they can be used to invoke a sub-dialogue between the computer and user, for example to allow the user to set specific options or load/save a file. This is usually part of a larger task that the user is performing. There are two main classes of dialogue boxes: Modal and modeless [7, 8]. Modal dialogues are described by Apple Computers ([7], pp 67-68) thus: “A modal dialog [sic] box is one that the user must explicitly dismiss before doing anything else, such as making a selection outside the dialog box or choosing a command”. This type of dialogue is used to present important information, as described above. The other type is the modeless dialogue, described thus: “A modeless dialog box allows the user to perform other operations without dismissing the dialog box”. This type of box is used to allow the user to supply option information, for example.



An error has occurred. Do you want to continue ?

Figure 6.2*: An alert dialogue box.*

Modal dialogue boxes

Modal dialogue boxes must be dealt with before any further interaction can take place. This is an example of a mode (as its name implies): Such boxes only allow the user to press the buttons in the box, any other characters typed or mouse-clicks will be lost. At first sight it seems unlikely that this mode will be missed because there is strong status feedback (see Figure 6.2). However, ‘hunt-and-peck’ typists usually look at the keyboard and copy-typists at a source document so they may not notice the box appear and so keep typing. Their keystrokes will then be lost. Therefore information is hidden due to the avoidable nature of visual feedback. The other problem is that users might forget the dialogue box is being displayed, for example the ‘cup-of-tea’ problem described above, and when they come back to their machine they start typing and keystrokes are lost.

Some potential errors that can occur when using modal dialogue boxes have been described. An ESM analysis will be performed to see if it suggests where the problems might be and what type of feedback could overcome them. This analysis is shown in Table 6.2.

Generic modal dialogue box

The system is in a mode when the modal-dialogue box is displayed; the user cannot do anything but deal with it. The mode will be action-independent (the mode only responds to OK/Cancel events not any other user actions, it will continue until the event dismissing the box occurs), demanding (the user must know he/she is in the mode), sustained (the mode will continue until the event dismissing it occurs) and static (the mode will not change the set of events and status that are permissible).

Generic modal dialogue box: **Mode** The system is in a mode as long as the box is displayed - user cannot do anything else. Box can be dismissed by clicking OK or Cancel buttons **Event** System error ➜ box displayed Box displayed ➜ OK/Cancel pressed ➜ no box **Status** Display of dialogue box	**Feedback:** **Mode** Action-independent, demanding, sustained and static **Event** Action-dependent, demanding, transient, static for both events **Status** Action-independent, demanding, sustained, static
Real modal dialogue box: **Mode** The system is in a mode as long as the box is displayed - user cannot do anything else. Box can be dismissed by clicking OK or Cancel buttons **Event** 1. System error ➜ box displayed 2. Box displayed ➜ OK/Cancel pressed ➜ no box **3. Erroneous key presses and mouse-clicks** **Status** 4. Display of dialogue box	**Feedback:** **Mode** Action-independent, **avoidable**, sustained and static **Event** 1. Action-dependent, **avoidable**, transient, static 2. Action-dependent, demanding, transient, static **3. Action-dependent, demanding, transient, static** **Status** 4. Action-independent, **avoidable**, sustained, static

Table 6.2*: ESM analysis of a modal dialogue box.*

There are two types of events. The first is the appearance of the modal dialogue box, this is often a system event, for example the system asks the user if he/she wants to save the current document after a time limit has passed. A dialogue box may also be displayed because an error has occurred in the system. It must be made a perceivable event for the user because he/she did not initiate it and so might not notice. In many situations the user directly and knowingly causes events and so they are perceivable, for example a mouse click - the user knows the mouse has been pressed because he/she actively had to press it. In the case of the modal dialogue the user may not directly cause the display of the box (for example, a time limit passed so that a save dialogue box is displayed) so it must be indicated saliently.

The second event is the dismissal of the dialogue by, for example, the user clicking the OK or Cancel buttons. This is a perceivable event for the user as he/she must actively click the screen button. Both of these events are action-dependent (the system must generate the event to bring up the box and the user must click the button to dismiss it), demanding (the user must know when the box comes and goes), transient (the event only lasts a short time but initiates a change in status) and static (it indicates one thing, the arrival of a dialogue box). When the modal dialogue box is on the screen the user cannot do anything but deal with it. The only acceptable events are the pressing one of the dialogue box buttons. All other events from the user will be rejected.

The status feedback about the dialogue box comes from the screen. The dialogue box appears on the screen when the event occurs and then continues as status information until it is dismissed by the other event. The feedback is action-independent (the box continues to exist regardless of user actions until the close event occurs), demanding (the user should not be able to miss the box, they should be forced to notice it), sustained (the box continues until the user dismisses it) and static (the box does not change, it just presents its message). As can be seen, the status feedback is the same as required to make the mode explicit.

Real modal dialogue box

The mode in the real dialogue is the same as the generic one. This time, however, the mode is avoidable. The discussion above showed that users might not notice the mode because they were not looking at the screen, for example hunt-and-peck typists. The visual nature of the feedback makes the mode avoidable.

The first two events are the same as in the generic dialogue box. The difference is that the appearance event is avoidable - there may not be a perceived event for the user because he/she is not looking at the screen and it may be a system initiated event. The user might be looking at the keyboard or their source document and keep typing so that keystrokes are lost. The dismissal of the mode will be demanding as the user has to click a button in the dialogue box to make it go away. The perceived event will occur at the same time as the actual event. There is, however, an extra event. Some systems provide additional feedback in the form of beeps or screen flashes if the user tries to type or click the mouse outside the box. These actions are not allowed in the mode and so cause error events. This action-dependent, demanding event feedback indicates to the user that the system is the dialogue box a mode.

Status feedback from the real dialogue box (see Figure 6.2) is avoidable, as is the nature of visual feedback, and users can miss it because they are not looking at the screen. The event changing the status often comes from the system and not the user. Therefore, the user is not likely to see the event or the change in status.

The auditory feedback on error events overcomes the problem with the avoidable screen feedback but does require the user to make errors before he/she perceives the mode. This may mean keystrokes are lost by the system and the user has to re-type them. If the user left the machine and came back to it later (as in Dix's ‘cup-of-tea’ problem), he/she would not be able to observe the state of the system because the auditory feedback is action-dependent. One way to get around this problem would be to have demanding action-independent status feedback. This was ruled out in the caps-lock key example above because that mode was not important enough to warrant the intrusiveness of the feedback. In this case, however, the modal dialogue box indicates an error or other important occurrence so constant, demanding feedback could be used. A sound could be played for as long as the mode existed. It could begin when the event initiating the mode occurred and continue as status information until the mode was dismissed. This would deal with the two problems described above. The user would know when the mode began and, if they left the machine and came back to it later (as in Dix's 'cup-of-tea' problem), they would know the mode was still in operation.

In some systems, such as the Apple Macintosh, modal dialogue boxes can be given an error level that describes the urgency of the problem the box reports. On the Macintosh there are three levels: Note, Caution and Stop [7]. These are usually differentiated by a different icon in the top left corner of the dialogue box. The icons contain: *, ? and ! for each of the levels respectively. It is often the case that users do not pay attention to the icon and level of the dialogue; they just deal with the problem it describes. If constant audio feedback was provided then this could be changed to reflect the level, or urgency, of the dialogue. As described in Chapter 3, Edworthy *et al.* [64, 65] have provided guidelines for controlling the urgency of sounds.

Modeless dialogue boxes

As described above, modeless dialogue boxes do not have to be dismissed before the user can carry on with other interactions. Such a dialogue box can be put into the background. The main problem is that if the box is the active window the user might not notice and so try and type into another window by mistake.

The modeless dialogue box is a mode, contrary to its name. The mode can be changed by selecting another window whereas in the modal dialogue only the OK/Cancel buttons could change the mode. In the real modeless dialogue box the mode feedback is avoidable: Users do not always notice that the modeless dialogue is not the active window when they try to use it. This is an example of an *unselected-window* error. This type of error is dealt with in greater detail in Section 6.6.7 on windows.

The problem occurs due to the avoidable status feedback from the active window. One way around this problem would be to provide demanding continuous feedback when the mode was active (as in the previous example). This would solve the problem as the user would immediately perceive the active window. This, however, could become annoying and the modal dialogue is not necessarily important enough to require this type of feedback. As in the caps-lock example, action-dependent, demanding feedback could be given to indicate the mode when the user typed and the mode was active. This would solve the problem but the user would have to type before he/she could tell the mode and thus errors might occur.

6.6.3 Buttons

One of the most fundamental widgets in all graphical interfaces is the button. These are isolated, individual regions within a display which can be selected by the user to invoke specific operations [58]. To avoid confusion, in this section *screen button* will be used to refer to the button on the computer display and *mouse button* will be used to refer to the button on the mouse. Dix *et al.* [57] and Dix & Brewster [61] have looked at some of the problems of screen buttons. The main one is that the user thinks the screen button has been pressed when it has not. This can happen because the user moves off the screen button before the mouse button is released. This is caused by a problem with the feedback from the screen button. If the user presses the screen button it becomes highlighted, if the mouse button is then released with the mouse still over the screen button it becomes un-highlighted and the operation requested takes place (see Figure 6.3). If the user presses the mouse button over the screen button, then moves (or slips) off the screen button and releases the mouse button, the screen button becomes un-highlighted (as before) but no action takes place. The feedback from these two different situations is *identical*.

Figure 6.3 shows an example of a correct selection and a slip-off. 1A shows the starting position with the mouse off the button. 1B shows the mouse button down on the screen button and the screen button highlighted. 1C shows the mouse button released over the screen button and the highlight removed. The interaction has completed successfully. In the second example, 2A shows the same starting position as before. In 2B the user has pressed the mouse button down over the screen button. In 2C the mouse has been moved off the screen button and the highlight has disappeared. The feedback is the same as 1C but the interaction does not complete successfully.

The identical feedback might not be a problem if the user was looking at the screen button but this is not the case. Dix & Brewster [61] suggest there are three conditions necessary for such slip-off errors to occur:

i) The user reaches closure after the mouse button is depressed and the screen button has highlighted.

ii) The focus of the next action is at some distance from the screen button.

iii) The cursor is required at the new focus.

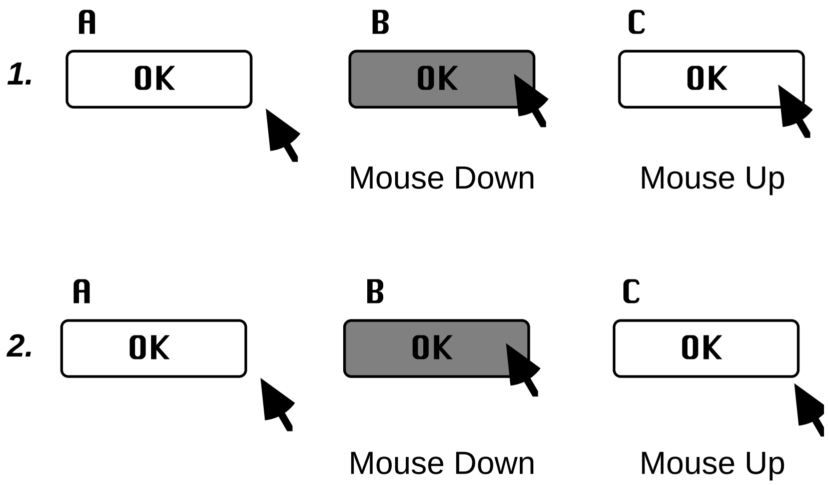


***Figure 6.3**: Feedback from pressing and releasing a screen button. (1) shows a correct button selection, (2) shows a slip-off.*

Because of closure (i) the mouse movement (iii) can overlap with the release of the mouse button - a slip-off. The attention of the user is no longer at the screen button (ii) so he/she does not see the feedback indicating there is a problem. The problem occurs because the actions become automatic and the expert user does not check each stage explicitly; it is an action slip.

These problems occur in screen buttons that allow a ‘back-out’ option: Where the user can move off the screen button to stop the action. If the action is invoked when the mouse button is pressed down on the screen button (instead of when the mouse button is released) then these problems do not occur as the user cannot move off. These screen buttons are less common (see Myers [120]) and can be more dangerous because users cannot change their minds. For a formal analysis of the different types of buttons see Bramwell & Harrison [28].

The event, status and mode analysis technique will now be used to see if it can suggest any solutions to the slip-off problem. Table 6.4 shows the ESM analysis of the problems with graphical buttons.

Generic screen button

The mode in this interaction begins when the mouse button is pressed down over the screen button and ends when the mouse button is released. This mode will still be in operation if the user moves the mouse off the screen button but still has the mouse button pressed down. This mode is action-independent (the mode continues until the mouse up event), demanding (this mode is kinesthetically maintained and so, as Sellen *et al.* showed above, is hard to avoid), sustained (the mode continues until the user releases the mouse) and static (the mode does not change). It is often the case that the press and release events occur very close together rather than the user holding the mouse button down, waiting and then releasing it. In this case the mode is only sustained for a short time and could be classified as transient. However, because it can be maintained for longer periods it is classified here as sustained.

Generic screen button: **Mode** When mouse button down in screen button (or moved off screen button) **Event** Mouse down ➜ mouse up Mouse down ➜ move off ➜ mouse up **Status** 1. Display of screen button 2. Highlighting of screen button	**Feedback:** **Mode** Action-independent, demanding, sustained, static **Event** Mouse down: Action-dependent, demanding, transient, static Move off: Action-dependent, demanding, transient, static Mouse up: Action-dependent, demanding, transient, static **Status** 1. Action-independent, avoidable, sustained, static 2. Action-independent, demanding, sustained, dynamic
Real screen button: **Mode** When mouse button down in screen button (or moved off screen button) **Event** Mouse down ➜ mouse up Mouse down ➜ move off ➜ mouse up **Status** 1. Display of screen button 2. Highlighting of screen button: Feedback when mouse button down in screen button, no feedback when release mouse button over screen button, no feedback when move mouse off screen button with mouse button still down	**Feedback:** **Mode** Action-independent, **avoidable**, **transient**, static **Event** Mouse down: Action-dependent, demanding, transient, static Move off: Action-dependent, **avoidable**, transient, static Mouse up: Action-dependent, demanding, transient, static **Status** 1. Action-independent, avoidable, sustained, static 2. Action-independent, **avoidable**, **transient**, dynamic

***Table 6.4**: ESM analysis of a screen button.*

There are two sequences of events. Pressing the mouse button down over the screen button and then releasing it over the screen button will be described first. In this case, the feedback is action-dependent (the user must press the mouse), demanding (the kinesthetic feedback is demanding, as shown above), transient (the mouse is only pressed for a short time) and static (the feedback does not change). The second series of events is where the user moves the mouse off the screen button before releasing the mouse button. In this case the feedback for moving off is action-dependent (the user must move off), demanding (the user must know they have moved off), transient (the event of moving off only lasts a moment) and static (the event causes a change in status but does not change itself). The perceived event should occur directly after the actual event in each of the cases as the user actively has to perform the actions causing the events.

There are two types of status information to display the mode. The first is the display of the button on the screen. This is action-independent (the display of the button continues whether there are actions or not), avoidable (the user should not be forced to perceive the button all the time), sustained (the feedback continues over time) and static (the button itself does not change). The second item of status information is the highlighting of the button. It is action-independent (the highlight only changes when one of the events occurs), demanding (the feedback should show that the screen button is being pressed or not), sustained (for as long as the screen button is pressed) and dynamic (the feedback changes depending on whether the button is being pressed or the user has moved the mouse of it). This status information changes when the events occur.

Real screen button slip-off

The mode is the same as in the generic example. However, the feedback from the real mode (see Table 6.4) is avoidable and transient. The user is not forced to perceive the feedback so can miss it.

The events are the same as for the generic button. The only difference is that the move off event is avoidable. The user might not notice that he/she has moved the mouse off the screen button. The status information is also the same as for the generic button but feedback from the highlighting of the screen button (2) is avoidable (users might not see the highlight because they have moved on to the next task) and transient (the highlight only lasts a short time so can be missed). There is not enough status information to make the mode explicit.

The avoidable feedback from the event of moving the mouse off the screen button combined with the avoidable, transient feedback from the button itself means that the mode is avoidable. This leads to the problem discussed above where the user may slip-off the screen button before releasing the mouse button and not notice. Auditory feedback could be provided to overcome this problem. Demanding, transient feedback could be given to indicate a successful press of the button, i.e. when the mouse is pressed and released over the button. An unsuccessful press should have no sound. This would mean that a successful and unsuccessful press would be displayed differently. Extra visual feedback could be given to do this but it would not solve the problem of closure (described above). Once the user presses the button he/she moves on to the next task which may be at some distance from the button. If visual feedback is given to indicate the two different types of button press the user might not perceive it as his/her visual attention would not be in the correct place. The advantage of auditory feedback is that it can be heard from all around so the user would perceive it and could tell if there had been a problem pressing the button. For a detailed experimental investigation of using sound in this way to overcome the problems of buttons see the next chapter.

6.6.4 Button palettes

Palettes of buttons also have problems. They are toggle switches where only one can be active at a time (like radio buttons). An example palette of buttons from HyperCard is shown in Figure 6.4. Palettes are often used in drawing packages to provide the user with different tools for operations such as drawing lines, circles or squares. In the diagram the pointer tool is selected. When another is chosen the pointer will be de-highlighted and the new tool highlighted. Each one of the tools is a mode, the system might be in square drawing mode or line drawing mode. As Dix *et al.* [58] say, palettes are a method of displaying the set of possible modes and indicating the active mode. The feedback is sustained so the user can see the current mode. One of the problems is that the feedback from the palette is avoidable. Palettes are often positioned at the side of the screen, away from the main working area and outside the users' area of visual focus. Therefore, status feedback about which is the active tool or mode is avoidable.

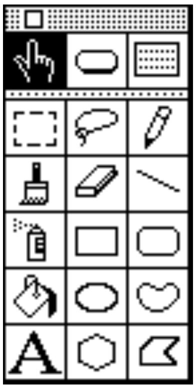


***Figure 6.4**: An example palette of buttons.*

One common error with this type of system is, for example, drawing a box using the box drawing tool and then wanting to select the box to move it to a new position. Unfortunately, the box does not move because the box drawing tool and not the pointer is the currently active mode. The user reaches closure on the box drawing task and then wants to position the box but forgets to choose the correct tool. The feedback is avoidable (because it is out of the focus of visual attention) when the user needs it to be demanding. To solve this the cursor could be changed to indicate the current mode at the users point of visual focus, for example by taking on the shape of the currently selected tool. One problem with this is that the cursor may obscure the work underneath if it becomes too big. It also has to compete for attention with the visual nature of the task being undertaken. Sound could be used to overcome the problems. Feedback about the currently active tool could be action-independent (the feedback continues until another mode is selected), demanding (so that the user always knows what mode the system is in), sustained (for as long as the mode lasts) and static (the feedback does not change until an event occurs).

As mentioned in the example of the caps-lock key above, action-independent, demanding sounds should only be used in important situations, for example the modal dialogue box. In this case, the situation may not be important enough to justify such potentially annoying feedback. Therefore, action-dependent feedback could be used instead. No auditory feedback would be given until the user pressed the mouse button, then feedback indicating the currently selected mode would be presented until the mouse was released. Whilst the user was drawing he/she would get mode feedback. When drawing was finished and the pointer tool required, the user would click to position the square he/she had drawn and hear the sound of the drawing tool and not the pointer tool. The extra feedback might provide the same advantages as that of Monk, described above. In his experiment with the extra mode feedback subjects knew which mode they were in: They did not make errors and then recover from them more quickly, they made fewer mode errors. Monk suggested that this was due to a better general awareness of modes in the system. In the case of button palettes, the extra mode feedback might remind users that they needed to change mode without having to first generate an error.

6.6.5 Menus

Menus are very similar to buttons. Menu items are selected by moving up or down a menu with the mouse button down and then releasing the mouse button over the required item. Users can back-out if they want to by moving off the menu but this facility can again can lead to slip-off errors.

The ESM analysis is very similar to the one for buttons above. In the generic menu, the mode occurs when the mouse button is down in the menu. This mode should be action-independent (it continues until the user releases the mouse button), demanding (the user should know he/she is in the menu), sustained (the mode is sustained for as long as the user has the mouse down) and static (the mode does not change, it allows the user to choose different menu items but the mode is the same: The events and status information are constant). The events are similar to the button example. The user can press the mouse button down over the menu and drag up and down until the item required is reached. The mouse button is then released. The mouse can also be dragged out of the menu and released. Feedback from these events is action-dependent, demanding (the user should know if he/she moves from one item to the next or out of the menu), transient (events last a short time) and static (the feedback from the event does not change). Status feedback comes from the menu. Dragging the mouse up or down changes the status feedback. The feedback should be action-independent (feedback is only changed by an event), demanding (the user should perceive which is the selected menu item), sustained (the feedback lasts as long as the menu is down) and dynamic (the status feedback changes in response to events).

The mode in the real menu is the same as the generic one but is avoidable. This means the user can slip off the menu without noticing, as in the button example above. The events are the same as the generic menu but the feedback from the move off event is avoidable. This is because the user's visual attention may be directed elsewhere. Real menus often use extra visual status feedback to differentiate a correct menu selection from an incorrect one. A successful selection has a multiple flash of the menu item selected before the menu disappears. An unsuccessful selection has no flash; the menu just disappears. This feedback differentiates a correct selection from an incorrect one. This feedback does not, however, solve the slip-off problem because it is avoidable due to closure. The user selects the menu item then reaches closure and moves on to the next task. He/she will not notice the lack of a menu flash as visual attention is directed elsewhere. If demanding auditory feedback was added (in a similar way to the button example) to indicate a successful menu selection event then users would again perceive it wherever their attention was focused. The slip-off problem would be solved.

Adding demanding auditory feedback to indicate a mouse-up over a menu item event would not solve a variation of the slip-off error that occurs with menus. If the user slips-off upwards or downwards from the menu item he/she wanted then another item is selected instead. This would still be a correct menu selection because the user did not slip-off the menu, so the correct selection sound would be played. The problem in this case is that the highlight from each of the menu items is identical and so becomes avoidable. To solve this, demanding tones of decreasing pitch could be added to each of the items in the menu. These would differentiate the menu items making them demanding. When the mouse button was pressed down in the menu an action-dependent, demanding, high-pitched tone would sound. This sound would be sustained until the next menu item was selected. As the user moved down the menu the tones would dynamically decrease in pitch. When the user arrived at the item he/she wanted its tone would sound and then the mouse button would be released over the item. The correct selection sound would be played after the tone for that menu item. If, for example, the mouse slipped on to the menu item above by mistake, a higher-pitched tone would be heard before the correct selection sound. This sound would indicate to the user that the wrong item had been selected.

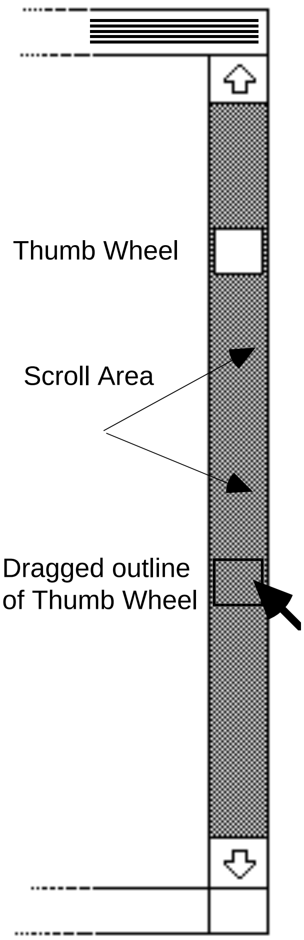


Figure 6.5*: An example scrollbar.*

6.6.6 Scrollbars

In the following example, scrollbars are examined using the ESM technique to find out if problems associated with them can be identified and characterised. Scrolling through a document can be achieved in several ways (see Figure 6.5). The user can click on the arrows at the top or bottom of the scrollbar, in the grey scroll area or drag the thumb-wheel (the white box in the scroll area). The thumb-wheel gives status information about the currently viewed position in the document. As Myers' [120] shows, scrollbars vary little in their general design. Therefore this analysis will be applicable to scrollbars in many different interfaces. There are three common problems associated with scrollbars:

❖ *Dragging the thumb wheel out of the 'hot spot':* The thumb wheel is the part of the scrollbar that allows the user to move by an arbitrary amount (the white box in Figure 6.5). The 'hot spot' is an invisible region around the scrollbar where the thumb can be dragged as if the mouse pointer was still inside the scrollbar. When the user has the mouse down and is dragging the thumb the system is in 'dragging' mode and status information should be given to indicate this. When dragging the thumb up or down in the document one may move too far to either the top, bottom, left or right of the scroll bar (out of the 'hot spot') and the thumb is lost. The document then does not scroll when the mouse is released and the user may become confused. When dragging only the outline of the thumb is moved, the real one stays where it is, and this does not grab the user's attention because it is harder to see - the information is hidden.

- *Position awareness in the document:* When scrolling through a document it can be hard to maintain a sense of position. The text can scroll by too fast to see (and the thumb only gives an approximate position). Some systems such as Microsoft Word put a page count in the bottom left hand corner of the screen but this is too far from the centre of visual focus so is hidden. One other method, used by MacWrite, is to put the page number in the thumb wheel itself. This is closer to the user's centre of visual focus and therefore should be more effective. The problem with this is that the thumb is very small so only a small amount of data can be displayed. It may also be missed by users if they are looking at the main part of the window and not the scroll bar. It may force users to look at the scrollbar when they really want to look at the screen.

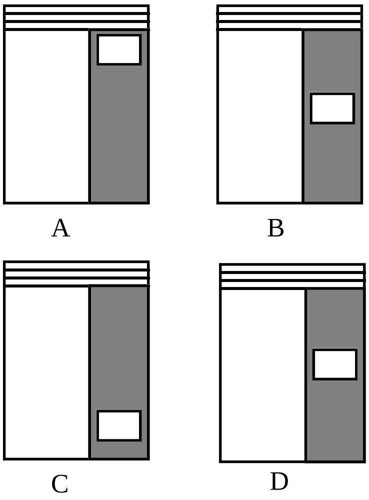


Figure 6.6: *Scrollbar 'kangarooing'.*

- *'Kangarooing' with the thumb wheel:* Repeatedly clicking in the scroll area above or below the thumb scrolls by a window-sized step. Clicking below the thumb scrolls down in the document and clicking above scrolls up. When the thumb is just above the target position (pointer location) it will scroll down to the window below (because the pointer is below the thumb) and then back up to the window above the target (because the pointer will then be above the thumb) and keep on doing this until the user notices and stops clicking. If the document is scrolling fast then it can be hard to tell this is happening as the user cannot see the text in the window (because it is moving to fast to provide any status cues) so the information is hidden. Figure 6.6 shows an example of kangarooing. In A the user begins to scroll down towards the pointer. In B the thumb wheel is just above the pointer. In C the user has clicked and the thumb has scrolled below the pointer. In D the user clicked again and the thumb scrolled back above the pointer so kangarooing occurred. Unless the user is looking at the thumb it can be hard to recognise that this has happened.

Table 6.5 and Table 6.6 show the ESM analysis of a scrollbar. The two different interactions were done in separate tables to avoid clutter.

Generic Scrollbar dragging: **Mode** When mouse button in thumb wheel **Event** Drag in 'hot spot' Drag out of 'hot spot' Press mouse Release mouse **Status** 1. Scrollbar thumb 2. Window	**Feedback:** **Mode** Action-independent, demanding, sustained, static **Event** Action-dependent, demanding, transient, static for all events **Status** 1. Action-independent, demanding, sustained, dynamic 2. Action-independent, demanding, sustained, static/dynamic
Real Scrollbar dragging: **Mode** When mouse button in thumb wheel **Event** 1. Drag in 'hot spot' 2. Drag out of 'hot spot' 3. Press mouse 4. Release mouse **Status** 5. Scrollbar thumb 6. Window	**Feedback:** **Mode** Action-independent, **avoidable**, sustained, static **Event** 1 & 2. Action-dependent, **avoidable**, transient, static 3 & 4. Action-dependent, demanding, transient, static **Status** 5. Action-independent, **avoidable**, **transient**, dynamic 6. Action-independent, demanding, sustained, static/dynamic

***Table 6.5**: ESM analysis of scrollbar dragging thumb out of 'hot spot'.*

Generic scrollbar dragging

The mode involved with dragging occurs when the mouse button is pressed down in the thumb wheel of the scrollbar. Moving the mouse will then drag the scrollbar thumb. The mode is therefore action-independent (the mode continues until the user releases the mouse), demanding (the user should know when they have the mouse button down and are dragging the thumb), sustained (as the drag will take some time) and static (the mode does not change).

There are two sequences of events: Pressing the mouse button down in the thumb, dragging it within the ‘hot spot’ and then releasing the mouse button; and pressing the mouse button in the thumb, dragging it outside the ‘hot spot’ and then releasing the mouse button. The feedback from both of these should be action-dependent (the user must actively drag the thumb), demanding (the user should know the drag is happening), transient (these are atomic events) and static (the feedback from the events does not change but they do change the status). The dragging events cause a change in the status feedback. At a lower level of abstraction the drag event is made up of smaller move events and status changes. The appropriate level of abstraction in this case is to consider a drag as an atomic event because all that concerns us here is where the mouse is dragged and released (in or out of the hot spot).

Status feedback comes from the scrollbar thumb and the window. Feedback from the thumb is action-independent (it gives status information no matter what the user is doing), demanding (the user should see the thumb because it is the primary means of interaction with the scrollbar), sustained (the thumb is displayed for as long as the scrollbar) and dynamic (the thumb move events change the status feedback). Feedback from the window can have two forms. In some systems, such as the Macintosh, the window does not scroll until the mouse button has been released. In other systems the window scrolls, in real-time, as the thumb is dragged. In the former case the window status feedback is changed by the release mouse event, in the latter case on the drag in ‘hot spot’ event. The feedback in the former case is action-independent (the window gives feedback regardless of user actions but is changed by a mouse release event), demanding (the user should know when the window scrolls), sustained (feedback from the window continues until the window is closed) and static (the feedback does not change until the mouse button is released). In the latter case the feedback is the same except that it is dynamic (the feedback changes in response to the drag event, rather than just the release event).

Real scrollbar dragging

In the real scrollbar the mode occurs again when the mouse button is down in the scrollbar thumb. Feedback from the mode is avoidable. There is not enough status information for the user to fully perceive the mode. Visual attention must be divided between the screen and the thumb wheel. If the user looks at the screen then he/she may miss information in the thumb and *vice versa*. In systems where the window updates when the mouse is released, dragging the thumb wheel out of the hot spot is a common error because the feedback from the dragging events (1 & 2 in the real scrollbar of Table 6.5) is avoidable. There is also avoidable and transient feedback from the scrollbar thumb. Only the outline of the thumb is dragged, the real thumb stays where it is (see Figure 6.5). This feedback is transient: When the user moves out of the hot spot, the outline of the thumb disappears. In systems that update on drag events this type of slip-off error would not occur because as soon as the user moved out of the ‘hot spot’ the screen would stop scrolling: The feedback would be demanding.

Systems have ‘hot spots’ to cope with variations when the user is dragging the thumb. They can vary in size from the whole width of the screen (the user can never move out of the hot spot by mistake), to a small area to the top bottom, left or right of the visually presented scrollbar or just the width of the scrollbar itself. In the first case users can never lose the thumb by mistake but they also cannot move out of the hot spot on-purpose to cancel the dragging. They must move the mouse back to where they started. Systems with very narrow hot spots (such as Microsoft Windows) allow users to easily cancel dragging but they can consequently very easily drag the thumb out of the hot spot by mistake. Therefore, having a medium-sized ‘hot spot’ around the scrollbar is beneficial. In systems such as this (for example, the Macintosh), the hot spot is not indicated on the screen, the only way a user knows it exists is indirectly because the thumb disappears when he/she moves out of it (the feedback is avoidable). To overcome this in sound, auditory feedback could be given when the user is dragging the mouse. When he/she moves the mouse out of the hot spot, the event could be marked by a change in the sound. The user would notice this demanding event sound. The new sound would be sustained until the user moved the mouse back into the hot spot or released it outside. This would overcome the problems described above.

An alternative approach could be used to minimise the potential annoyance of the sustained auditory feedback. Sound could be played as the user drags within the hot spot but when he/she moves outside the sound stops. The cessation of sound would be a demanding event in itself and the user would know that he/she had left the hot spot. The feedback would be transient (as the sound stopped) and dynamic. This method would reduce the amount of sound feedback necessary and so avoid the possibility of annoyance (for more of a discussion on the problems of annoyance see the next chapter).

Generic scrollbar kangarooing

Table 6.6 shows the ESM analysis of scrollbar kangarooing and loss of position. As in the scrollbar dragging example, the mode occurs when the mouse button is clicked in the scroll area. A single click of the mouse scrolls by one window of data. Holding the mouse button down (or clicking the mouse many times) continuously scrolls by multiple windows of data. The mode should be action-independent (the mode continues until the mouse is moved out of the scroll area), demanding (users should know that they are in the mode), sustained/transient (the mode can be sustained, if the user keeps the mouse button pressed down, or transient if the user just clicks the mouse) and static (the mode will not change until the user moves out of the scroll area).

Generic scrollbar kangaroo: **Mode** When mouse button clicked in scroll area **Event** 1. Click in scroll area 2. Thumb reaches target 3. Cross page boundary **Status** 4. Scrollbar thumb 5. Window 6. Position in document indication	**Feedback:** **Mode** Action-independent, demanding, sustained/transient, static **Event** Action-dependent, demanding, transient, static for all events **Status** 4. Action-independent, demanding, sustained, dynamic 5. Action-independent, demanding, sustained, dynamic 6. Action-independent, avoidable, sustained, dynamic
Real scrollbar kangaroo: **Mode** When mouse button clicked in scroll area **Event** 1. Click in scroll area 2. **No thumb reaches target event** 3. Cross page boundary **Status** 4. Scrollbar thumb 5. Window 6. Position in document indication	**Feedback:** **Mode** Action-independent, **avoidable**, sustained, static **Event** 1. Action-dependent, demanding, transient, static 3. Action-dependent, **avoidable**, transient, static **Status** 4. Action-independent, **avoidable**, sustained, dynamic 5. Action-independent, demanding, sustained, dynamic 6. Action-independent, avoidable, sustained, dynamic

Table 6.6: *ESM analysis of scroll bar loss of sense of position and ‘kangarooing’.*

There are three events. The first is the user clicking in the window scroll area, the second is the thumb reaching the target location and the third the crossing of a page boundary when scrolling. Feedback from the click event should be action-dependent (the user must press the mouse button), demanding (the user actively has to click the mouse), transient (the mouse is only pressed for a short time) and static (the feedback indicating the click does not change). When the thumb reaches the target (the cursor position) a demanding event should be given to alert the user. This event feedback should also be action-dependent (the thumb reaches the cursor position because the user clicks), transient (the event lasts a short time) and static (the feedback does not change). The crossing a page boundary event occurs when the user scrolls passed a page boundary. The feedback should be action-dependent (the user must click the mouse), demanding (the user should know when a page boundary has been crossed), transient (crossing the boundary lasts a short time) and static (the feedback just indicates that a boundary has been crossed).

There are three types of status information: Information from the thumb and when it moves, information from what is in the window and when that changes, and the position in document indication. This may be, for example, a page count in the thumb or elsewhere in the window. These three types of status feedback are changed by the user clicking in the scroll area. This event causes a change in the status of the window and scrollbar thumb. It may change the position in document indicator if a page boundary is crossed. Thumb wheel feedback should be action-independent (it gives information about position in document no matter what the user is doing), demanding (because the thumb is the primary means of interaction with the scrollbar), sustained (feedback from the thumb lasts for as long as the scrollbar is displayed) and dynamic (the feedback can change because of the click event). The feedback from the thumb should be demanding because it is the main source of feedback from the interaction. It gives information about where the user is in the document.

Status feedback from the window is similar. It is action-independent (it gives feedback no matter what the user is doing, although this changes when event 1 occurs in Table 6.6), demanding (it takes up a large area of the screen), sustained (the window lasts for a period of time) and dynamic (event 1 will cause the status information to change). The status feedback from the window is classed as demanding because it takes up a large area of the screen. The user can, of course, not look at the screen but if he/she is looking at it then the window is likely to be the area of focus. Event 1 will cause the window to scroll which is a very demanding occurrence because a large area of the screen changes.

The position in document indication should be action-independent (the feedback is given no matter what the user does), avoidable (the user should be able to sample the status information to find out what the current page is), sustained (the feedback lasts for as long as the window is displayed) and dynamic (the feedback may change in response to event 3). The page indicator changes its feedback in response to the crossing of a page boundary. The status feedback should be avoidable: Users will not want the information forced on them, they will want to be able to sample it if necessary. The page boundary event should, however, be demanding because the user should know when a new page has been reached so that they do not lose their sense of position in the document.

Real scrollbar kangarooing

In the real scrollbar, the mode is the same as above. The feedback on the mode is avoidable and this means that kangarooing and loss of position can occur. There is no event to indicate that the thumb has reached the target, which is why kangarooing can occur. This event information is hidden: The user will not know that the thumb has moved to the target position (usually the pointer location in the scroll area) unless he/she happens to be looking at the thumb. The event feedback indicating when a page boundary has been crossed is also avoidable. The user will not see the position in document indicator change because it is outside the area of visual focus. Often a dotted line is displayed in the window on the screen to show a page boundary. If the user is scrolling rapidly through the file then this too is likely to be missed.

The status information is the same as before but feedback from the thumb is avoidable: It is easy to avoid seeing the thumb move because it is small. The same is true of the position in document indicator (it is again avoidable) as it is out of the area of visual focus. The status feedback from the window is very demanding and when it changes (due to the click event) the change captures the user’s attention. Kangarooing occurs because the event indicating when the thumb reaches the target location does not exist in the real scrollbar. This is made worse because status feedback from the thumb is avoidable. If this was demanding the user would see when the thumb reached the target so kangarooing could be avoided. Loss of sense of position occurs because the page indicator is outside of the area of focus and the document can scroll by very quickly. The very demanding nature of the scrolling window causes the user not to look at the page count.

Sound could be used to overcome the problems associated with kangarooing. If a demanding sound was added to indicate the event of the thumb reaching the target location then the user would hear it and stop clicking the mouse. For example, if a low-pitched sound was played when scrolling down through a document and a high-pitched one when scrolling up then kangarooing would be perceived as an out of sequence high or low tone.

The crossing of a page boundary event could be given a demanding sound so that users knew when a new page had been reached. This would help them keep a sense of position in the document. The position in document indicator could itself be given an avoidable, action-independent sound. The user could then perceive the page without taking his/her visual attention away from the primary task. This could be done by having a sustained, dynamic tone, the pitch of which could change as a page boundary was crossed. This feedback would not be annoying, even though it was sustained, because it would not need to be demanding: It could be made to fade into the background of consciousness. The user would only perceive it when he/she wanted to listen for the page sound or when a page boundary was crossed. It would also indirectly give information about the thumb and thumb position. This information could also be added to the thumb wheel dragging sound described above. When the user dragged the thumb over a page boundary, the dragging sound could change in a demanding way to indicate a new page. Using this method it would be easy for the user to move two pages through the document, for example: He/she would just listen for two page boundary sounds. This analysis of scrollbar kangarooing and loss of position is experimentally investigated in the following chapter and was published in [35].

6.6.7 Windows

There are many different errors that occur in modern graphical interfaces but one of the most common is the ‘unselected-window’ error. This occurs in multi-window systems where the user tries to interact in one window but another is actually the selected, active one. This may cause keystrokes to be lost or errors to occur because the selected window is interpreting commands not meant for it. This type of error is an ‘action-slip’ (see section 6.2.2 above), as Lee [105] describes (p 73): “…as a skill develops, performance shifts from ‘closed loop’ control to ‘open-loop’ control, or from monitored mode to an, automatic, unmonitored mode of processing.” As the user becomes familiar with the task he/she no longer monitors the feedback so closely. This type of error is similar to that which occurs with the modeless dialogue box (see section 6.6.2 above). This box can be in the foreground or background. The user can mistake which of these is its real state and try and interact with another window when the box is active, or *vice versa*.

Harrison & Barnard [90] and Blandford, Harrison & Barnard [21] suggest that the problem is due to *interactional disengagement:* The user and system are engaging in different interactions. The user is trying to type into one window, the system is accepting data into another; so the user’s interaction is misdirected. Harrison & Barnard suggest two conditions for this to occur:

- ❖ On resumption after a pause the user’s focus has changed so that he/she is no longer ready to engage in the interaction the system is expecting. This is the ‘Cup of tea’ error [59]. For example coming back to the system after having gone away for some time, perhaps to have a cup of tea, the user must re-orient to the system. Often the first task to be done will cause the user to re-orient to that window because it is a priority, whereas another window is really the active one.

- ❖ Discontinuity of input, for example when the mouse is knocked into another window by mistake. The user still thinks the old window is active but it is not. Another example is a user activates another window to look at some data, but then fails to re-activate the main window again when switching attention back to it.

The user fails to observe the state of the system because the status feedback from the active window is avoidable. The predicted effect of the user’s commands is then incorrect and errors occur. Reichman [143] also investigated the unselected window problem and came to similar conclusions as Harrison & Barnard. She suggested that it was caused by a mismatch between user and system representations of the task. She noted that the error frequently occurred when moving from one window to another. For the user the windows were linked due to the nature of the task. To the system, the windows were totally separate and independent from each other. Reichman suggested that this conforms to a lack of shared contextual knowledge between the system and the user; there is no indication of the underlying window/activity interrelation. Harrison & Barnard suggest two solutions:

❖ Force the user to choose the application required before he/she can interact with the system.

❖ Reinforce the feedback from the active window so that the user does not mistake it.

Most attempts to solve the problem take the latter course. As Reichman says (p 296): “Most of these systems mark the active window by a little blinking pointer or a highlighted title bar. These visual indicators are too limited”. She suggested one solution to the problem: Use colour to reflect status (thus reinforcing feedback from the active window). She used green (Go) for the active window and red (Stop) for the inactive ones. This idea was dismissed because of problems with the colours having different meanings to different user groups. The work of Sellen *et al.,* described at the beginning of this chapter, would suggest that this might not solve the problem anyway due to conflicts with the visual task being performed by the user. She then suggested that shading could be used. The active window would be white and the other, inactive windows would have increasing levels of grey dots making them more and more difficult to read. There are problems with this method. Users must differentiate the different levels of grey [105]. The more windows there are the smaller the differences between them. Another problem is that users may not be able to see information in the other windows as they are greyed-out. One advantage of window systems is that users can see this information. For example, when writing a paper in the current window, one might also want a window of experimental results data in a spreadsheet and a window of a copy of another relevant paper. If these were greyed-out and they could not be read an important advantage of graphical interfaces would have been lost.

The ESM analysis of unselected-window errors is shown in Table 6.7. The solutions to the problem suggested from the ESM analysis described below fall into category of ‘reinforcing feedback’. The sounds used will attempt to make the feedback from the active window demanding.

Generic window: **Mode** The active window is the current mode **Event** 1. Make active 2. Make non-active **Status** 3. Active Window 4. Non-active window	**Feedback:** **Mode** Action-independent, demanding, sustained, static **Event** Action-dependent, demanding, transient, static for both events **Status** 3. Action-independent, demanding, sustained, static 4. Action-independent, avoidable, sustained, static
Real window: **Mode** The active window is the current mode **Event** 1. Make active 2. Make non-active **Status** 3. Window highlight 4. Non-active window	**Feedback:** **Mode** Action-independent, **avoidable**, sustained, static **Event** Action-dependent, demanding/**avoidable**, transient, static for both events **Status** 3. Action-independent, **avoidable**, sustained, static 4. Action-independent, avoidable, sustained, static

Table 6.7: *ESM analysis of unselected-window error.*

Generic window

The window has events that it responds to and presents status information about itself. Events in one window will be different to those in another. Different status information will be given from each window. There are two different types of windowing systems. One type requires the user to click in a window to make it active, the other just requires the user to move the mouse into it. In both cases the mode is action-independent (the mode continues until the user selects another window), demanding (the user must know when they have moved/clicked), sustained (the mode will continue until the user moves the mouse into another window) and static (the mode will not change). The mode continues until the user moves to or clicks in an other window.

There are two events: Make active and make non-active. These can be achieved by the two methods described in the previous paragraph. The feedback from both of these events should be action-dependent (the user must move/click the mouse), demanding (the user should know when a window is activated/deactivated), transient (the change window event lasts a short time) and static (the feedback only indicates one thing: The change from active to non-active or *vice versa*).

The status feedback in this case is from the highlight of the active window and the un-highlighted ones. The feedback from the active one is action-independent (the feedback from the window continues no matter what the user does although it is changed by events 1 or 2 in Table 6.7), demanding (the user should know which is the active window), sustained (the feedback continues until the user selects another window, event 2) and static (the feedback does not change). The only difference in the non-active windows is that the feedback from them should be avoidable, users can sample the information in a non-active window if they require but it should not be forced upon them.

Real window

In the case of the real window, the window is the mode as before but the feedback from it is avoidable. This means that users are not forced to know what mode, or window, they are in and so unselected window errors result. The mode is avoidable because of the events and status that comprise it. The make active/non-active event is avoidable. The user can knock the mouse into a window by mistake and then this becomes the active one, but there is no demanding event to tell him/her that this has happened. This type of problem is not one that occurs often in ‘click to select’ window systems such as the Macintosh. In this type of system, the click event is demanding because the user has to actively press the mouse to cause the event.

The status feedback from the window is also avoidable which means users do not notice which is the active window when they look at the screen. This is an example of a problem with actual and perceived status (as described in the section on status above). The user does not perceive the status information indicating which is the active window. This is because often only a small area around the edge of the window is changed to differentiate active from non-active windows. Lee [105] showed that if demanding, dynamic visual feedback was added to window borders (he used a ‘fizzy’ border) then the number of unselected window errors could be reduced (see Chapter 7 for more on Lee’s experiment). This feedback problem leads to the ‘cup of tea’ error described above. The user comes back to the computer after doing something else. The feedback from the active window is avoidable so he/she does not notice which is the active one and begins to type in the window that is thought to be active and then errors occur.

For the problem of avoidable event feedback an action-dependent, demanding, transient and static event sound could be added to indicate when the active window was changed. The sound would be played when a new window was made active. This would indicate to the user that the mouse had been knocked into another window by accident. For the problem of avoidable status, more demanding feedback is needed from the active window. It could be action-independent and demanding but problems of annoyance would mean that this method, although effective, might be unusable. Action-dependent, demanding feedback could be used instead. This would require the user to start interacting with the system but he/she would then hear a sound to indicate which was the active window. This might be a problem because users would potentially have to make errors before they could tell which was the active window.

One other way to solve the ‘cup-of-tea’ problem would be to add action-independent, avoidable, sustained and static feedback. For example, in the form of a quiet background tone, just above the ambient sound threshold. This would avoid the annoyance problem as constant tones have the ability to be habituated, or fade into the background of consciousness [39]. This would complement the action-dependent sound described above. Each application could be given its own timbre, the user would then learn the timbre associate with each application. If the user left the system and then came back to it later, he/she would not have habituated the sound (as it was not heard whilst the user was away from the machine) and so would hear it and recognise the active window without having to make any errors. This analysis is experimentally tested in greater detail in the next chapter.

6.7 A REVIEW OF THE ESM ANALYSIS TECHNIQUE

The above detailed ESM analyses have shown the method in practice. What conclusions can be drawn from them? The technique gives the interface designer a high-level, structured way of thinking about and analysing interactions. The problems with each of the widgets can be identified and audio feedback suggested that could overcome them. The technique is easy to apply and no knowledge of formal system models is necessary in order to use it. The technique can also be used to minimise the amount of audio feedback necessary to reduce the potential of annoyance.

The analyses showed that sound feedback is appropriate at the interface because some problems cannot easily be corrected with extra graphical feedback. This is due to the nature of the visual system. In the case of the screen button slip-off problem, users saw the highlight of the button and then moved on to their next task: They reached closure on the button press. If they slipped off the button they would not notice because their attention was elsewhere on the screen. If, to try and overcome this problem, extra graphical feedback was used (for example, making the feedback different when the user slipped-off to when a correct selection was made) this would be outside the small area of visual focus so the user would not see it. It would therefore be very difficult to signal this information graphically. Sound is the natural method of doing this because it is omni-directional. Sound also does not conflict with the other graphical information that the user might be working on. This shows the benefits that could be gained from exploiting the inter-relationships between graphics and sound: Both have different properties and when both are used together then they are very powerful.

The following three sections will describe the general conclusions drawn from the analyses about each of the three types of information. The characterisation will then be discussed and this is followed by a discussion of how the analyses fit with the types of hidden information.

6.7.1 Modes

In almost all of the examples mode information was hidden because of the avoidable nature of visual feedback. Modes were hidden because there was not enough status feedback to make them explicit. Modes were always action-independent. They were initiated and terminated by events but continued regardless of other user actions. For example, in the button press example the system was in the mode for as long as the mouse button was pressed down. When the mouse button was released the mode ended.

Problems occurred when modes were avoidable because of a lack of demanding visual feedback. If all of the status feedback in a mode was avoidable then the mode was avoidable. Modes were always sustained and static. They were sustained because they continued for a period of time (this was sometimes short in the case of a button press) unlike events which were discrete points in time. Modes were static because they did not change whilst they were in existence. They always allowed the same set of events and provided the same types of status feedback.

6.7.2 Events

Generic event feedback was the same in all of the examples: Action-dependent, demanding, transient and static. Events were always caused by an action on the part of the user or the system. They were always demanding because they marked important happenings in the system. They always occurred at discrete points in time and they did not change so were static. The main problems with events arose when they were not presented in a demanding way. In the case of the window described in the previous section, the user could knock the mouse into another window and that event would cause the new one to become active. The feedback from this was avoidable so that the user would not know the active window had changed.

In most cases the user-initiated events were demanding: The user had to actively click the mouse. In the case of the window and the caps-lock key user-initiated events could be avoided. All the events were user-initiated input events apart from the appearance of the modal dialogue box which was system-initiated (another example, not discussed here, is mail arrival). There was an important difference between these two types of events. The system initiated ones were avoidable because the user took no part in generating them. This meant care was needed to make sure that the actual events were perceived by the user.

6.7.3 Status

Status feedback showed more diversity than that from events. Status feedback was always action-independent (in the same way as mode feedback). It continued regardless of user actions until one of the events in the mode occurred and changed it. For example, the display of the modal dialogue box continued until the OK/Cancel event occurred.

In all of the examples, avoidable status information caused errors and led to modes being hidden. The main reasons status information was hidden was because of the nature of visual information. Users had to divide their visual attention between different areas of the screen and so might not be looking at the right place at the right time. It is often the case that visual information is avoidable because the user might be looking at the keyboard or a source document. In the example of the caps-lock key users did not know that it had been engaged because they were not looking at the screen. Status feedback could be both demanding and avoidable. In some cases (such as the modal dialogue box) the status feedback had to be demanding so that the user did not forget he/she was in the mode. In other cases it could be avoidable, for example background windows: The user could look at the window if he/she wanted to.

Sometimes status feedback was hidden because it was transient. Button slip-off and scrollbar dragging problems were caused by this. The user might not see the visual feedback because it only lasted a short time and therefore was avoidable. In the case of dragging in the scrollbar, when the user moved out of the ‘hot spot’ the status feedback from the thumb disappeared (was hidden).

Status feedback could be dynamic or static. In the case of the highlighting of the screen button, feedback was dynamic because it changed when the user moved the mouse into or out of the button. In the case of scrolling, the feedback was dynamic because it changed rapidly in response to move events.

6.7.4 Hidden information

Most of the reasons given at the beginning of the chapter for why information is hidden occurred in these examples. Status information was not available in the case of the button slip-off problem. In this example problems occurred because the feedback from a slip-off was identical to a successful selection. There was no indication to the user that an error had occurred. In terms of events, there was no event to indicate that the scrollbar thumb had reached the pointer location. This was one of the main reasons for kangarooing.

Visual feedback was difficult to access in many of the examples because of the avoidable nature of such feedback. The user did not have to look at the screen so he/she could miss feedback. In the kangarooing scrollbar problem, the visual feedback from the window was so demanding that it caused feedback from the thumb wheel or page count to be missed; the user was overloaded with too much visual information. He/she could not look at the thumb and page counter as well as the screen.

There were very many examples of the small area of visual focus causing problems. In the scrollbar kangarooing example both the thumb and page counter were outside the area of visual focus and so became avoidable. The user may also be looking at the wrong place on the screen and so not see the visual feedback. In the button slip-off example users were not looking at the button when it was indicating an error because they had reached closure. They were looking at the location where their next interaction was taking place.

In the case of the window the status feedback was avoidable because only a small amount of screen space was used to indicate the current window (just a change in the screen border). Even if more screen space was used to present status information it could still have been avoidable because the user does not have to look at the screen and the visual nature of the feedback conflicts with the task (see Sellen *et al.* above). The final reason, given at the beginning of the chapter, for information being hidden was due to modes. The examples here have shown that modes hide information because in many cases there was not enough status feedback to make the mode explicit.

6.7.5 Characterisation

One of the reasons for carrying out the analyses was to discover which of the feedback dimensions fitted with which type of information. For example, were events always demanding? Was status always sustained? The three types of information have been shown to fit well into the characterisation (see sections 6.7.1 to 6.7.3). In the analyses carried out here, for example, generic events were always the same but status could vary. Designers using the analysis technique could compare their analyses with those described here. If in one analysis events were shown to be different to the events described here then further investigation would be needed. It might be that their classification was wrong or that under different circumstances event and status feedback might be different. The characterisations described here could be used as a starting point.

How well did the characterisation indicate the feedback required? The examples of the event, status and mode analysis method described have shown that the action-independent/action-dependent dimension was not a useful one in this case. In all cases this did not vary: Modes and status were always action-independent whilst events were always action dependent. This category was only useful when converting from action-independent to action-dependent feedback to avoid annoyance.

In several of the examples above, action-independent, demanding, sustained and static status feedback was required (for example the caps-lock key or the modal dialogue box). This was not always appropriate because of the annoyance that this type of sound might cause. In the case of the modal dialogue box this feedback was acceptable because the box indicated an important error in the system so the user should be informed of this until it had been dealt with. In the case of the caps-lock key, this was not an important enough mode to require such potentially annoying feedback. In this case, the feedback could be changed to be action-dependent, demanding, transient and static to occur on a key press event. This would be less annoying for the user but would have the disadvantage that the mode might not be perceived until the user typed and key presses might be lost. This change from action-independent to action-dependent feedback was used in other examples to remove potential annoyance problems.

As described above, mode feedback was always sustained and event feedback transient. Status feedback could be either. Problems occurred when status feedback was transient as users did not always have chance to see it and therefore the information was hidden. For example, in the screen button example status feedback about the highlight of the screen button only lasted a short time. If the user moved off the screen button there was no status feedback about the highlight.

It was not always clear whether status feedback should be classified as dynamic or static. Most status feedback will change given enough time. In the case of the screen button, feedback only lasted a short time if users pressed and released the mouse quickly. As an alternative they could hold the mouse button down before they released it and the feedback could be sustained. The problem is: How long does the feedback have to be maintained before it is classified as sustained? This research classified the button feedback as sustained because it could be maintained under some circumstances. Further examples would need to be studied to find out more about this.

The most important dimension was the avoidable/demanding one. In many of the situations where hidden information occurred it was due to the information being avoidable when it should have been demanding. However, the demanding versus avoidable dimension was also not black-and-white. As was shown in the modal dialogue box example, different levels of ‘demandingness’ or urgency might be required. Feedback for important errors would need to be very demanding whereas for warnings less demanding. It was suggested in the dialogue box example that different types of sounds could be used to represent this.

Another facet of the demanding versus avoidable dimension was identical feedback. In some cases feedback from two situations was identical (for example in the button slip-off problem or menu selection). The feedback then became avoidable because the user could not tell the two situations apart. Further investigation might show that this could be split off into a separate dimension of its own because it is outside the main definition of what was avoidable.

6.7.6 The ESM analysis as part of the structured method for integrating sound into interfaces

The ESM analysis technique has been described in detail and has been applied to seven interactions to find where problems occur. It has shown where sounds should be added and suggested the types of audio feedback necessary to overcome the problems. These suggestions can be combined with the earcon guidelines from the previous two chapters to generate the sounds required. The technique has been shown to be easy to use and effective.

The work described here has substantially improved upon the method put forward by Dix to optimise it for finding where in the interface to use sound. Mode information has been included into the analysis. This is important as these are a major cause of errors in the interface. It also improves upon previous work by adding the categorisation of feedback. This means that along with identifying where hidden information exists the technique can suggest the type of auditory feedback needed to present it. Designers now have a tool that will allow them to add sound to an interface in an effective, clear and consistent way.

6.8 FUTURE WORK

The event, status and mode analysis technique should be applied to more interactions to find out where potential problems with hidden information occur. The interactions tested here all had known problems and the technique was applied to find out what was causing them. New interactions should be examined to see if the technique can find new problems and suggest solutions. This might cause the generation of more feedback dimensions than exist at present. When testing new interactions (where the problems are not known) different aspects of the analysis may become more important. This will make the technique stronger.

The dimensions of feedback used could be extended from just two opposite ends of a scale to a wider range. As was suggested in the demanding/avoidable dimension described above, a further breakdown might be useful to deal with more or less demanding feedback where the characterisation into one form or another is not black or white. Most of the problems in the interactions analysed were due to avoidable feedback. This dimension should be investigated further to see if it could be broken up into more categories to specifically pinpoint what the problems are.

The technique could be extended to use a more structured notation for describing the interactions. One such system is User Action Notation (UAN) [93, 94]. UAN considers small snippets of user behaviour, for example the deletion of a file. It describes the actions the user must perform and the feedback from the system. The notation can represent things such as moving the mouse, selecting icons, pressing and releasing the mouse button and various forms of highlighting. UAN could be used to make the descriptions of events and status in the left-hand side of the ESM analysis tables more precise and consistent across analyses. UAN is also easy to learn and does not require formal interface analysis skills. Using it would still mean that the analysis technique could be used by all interface designers but would provide more precise description of interactions.

6.9 CONCLUSIONS

This chapter put forward a method for finding where sound should be used in the human-computer interface. Prior to this work, there was no other method for doing this. Designers who added sound to interfaces had do it in an *ad hoc* way. This often meant that the sounds were ineffective and not consistent across applications. The analysis technique described here provides a structured way of modelling interactions to find where the hidden information that can cause errors exists and for categorising it. The technique also suggests the audio feedback needed to fix the problems. This is an important step forward in the adding of sound to user interfaces. Sounds no longer have to be added in an *ad hoc* way.

This analysis technique forms half of the structured method for adding sound to human-computer interfaces. The guidelines from Chapters 4 and 5 form the other half. A designer could use the analysis technique to identify aspects of the interface where hidden information is a problem. The suggested solutions to the problem can be used along with the earcon guidelines to create earcons to make explicit the hidden information. Using the method a designer who had no knowledge of sound design could produce a set of effective and usable sounds for an interface.

The technique showed that using more visual information to overcome various problems of hidden information would not necessarily work. Visual information is, by its nature, avoidable: A user might choose not to look at it. In some cases more visual feedback cannot solve the problems. For example, with screen buttons the user is no longer looking at the button when it gives error information because of closure; attention has moved to the next interaction and the previous one will be outside the area of visual focus. Sound can overcome this because it is omni-directional. This demonstrates the benefits that could be gained from exploiting the inter-relationships between graphics and sound: Both have different properties and when both are used together then they are very powerful. The technique also takes account of possible annoyance due to sound (such as with continuous demanding feedback) and can minimise it.

The technique has made predictions about where sound should be used. The only way to find out if the predictions made will actually overcome the problems of hidden information is to test them. The following chapter does just this to find out if the event, status and mode analysis predictions work in practice. The structured method for adding sound will be tried in full. The event, status and mode analyses will be used along with the earcon guidelines to sonify some interactions.

CHAPTER 7: SONICALLY-ENHANCED WIDGETS

7.1 INTRODUCTION

This chapter brings together the rest of the work described in the thesis. Three sonically-enhanced widgets were designed and tested. The work on earcons was used to add the sound to the widgets and the sounds were added in the ways suggested by the event, status and mode analysis technique described in the previous chapter. These two pieces of work together form the structured method for integrating sound into user interfaces. Chapters 4 and 5 showed that complex information could be communicated in sound. Chapter 6 suggested some areas where sound could be added to overcome problems of hidden information in many common graphical widgets. In order to test if the ways of adding sound suggested by the event, status and mode analysis technique were effective some sonically-enhanced widgets needed to be designed and tested based on the technique's predictions.

Adding sound in the ways suggested here should prove effective because it will give users information that they need but was previously hidden from them. Often sounds added to interfaces are just gimmicks. They do not give the user any new information but just redundantly present information already there. They are not designed as a coherent whole and are often too loud, annoying other users nearby. The sounds suggested here provide new information, are carefully designed to be coherent and consistent and care is taken to make sure they are not annoying.

The experiments described in this chapter directly test sonically-enhanced widgets against visual ones to see the effects of sound. The research will give evidence, by direct comparison of sound and graphics, to show that sound-enhancement has advantages over simple visual presentation.

This chapter begins with a discussion of annoyance due to sound and how it can be avoided. Annoyance was specifically investigated as part of the work in this chapter because it is seen as a problem by many potential users. The following section describes an experimental testing framework designed so that the sonically-enhanced widgets could be tested. This framework was used in the investigation of each of the widgets and provided a consistent method of testing the new widgets. The next three sections describe experiments to test sonically-enhanced scrollbars, buttons and windows. The experiments are described and the results discussed. The chapter ends with a general discussion of earcons based on the knowledge gained from implementing the sonically-enhanced widgets.

7.2 ANNOYANCE CAUSED BY SOUND FEEDBACK

One of the main concerns potential users of auditory interfaces have is annoyance due to sound pollution. This is closely linked to the problem of loudness described in Chapter 3. Sound is public whereas graphical output is private. When discussing the use of sound at the interface it is often suggested that it will be an annoyance. There are two aspects to this: It may be annoying to the user whose machine is making the noise and annoying to others in the same environment who overhear it. Buxton [38] has discussed this and shown that there is no such thing as absolute silence anyway and we all have to put up with varying amounts of noise in our working environments. He suggests that some sounds help us (information) and some impede us (noise). We need to control the sounds so that there are more informative ones and less noise. Sounds at the interface should be used in ways that provide information to the user. If they do this effectively then they will be informative; the user will want the information they contain.

The annoyance of sound is most often due to excessive intensity [18, 166]. Auditory feedback from most computers is too loud. Computer users normally sit very close to the terminal/workstation that they are working on so that quiet sounds can still be heard. Think, for example, of the noise of a hard disk that can be used as a check on activity within a system. Sound output from the computer should be at a level just above the threshold of background noise. These sounds will be audible but not intrusive. There are problems with the sound output hardware of many computers that make this difficult.

The speaker is often of poor quality and mounted deep within the case of the machine (often near the fan). This can lead to problems of localisation (see Chapter 2). Users can find it difficult to tell if a sound came from their machine or a colleague's because of distortion due to bad positioning of the speaker. Another problem is that of controlling intensity; very crude controls are usually available that do not allow the sounds to be turned down to a low enough intensity level. If more sophisticated sound output hardware were available then some of the problems due to intensity could be solved. This could be achieved by adding an amplifier and external speakers to the machine. This would also solve the problem of localisation.

Of course, one person's informative sounds are another's noise. If sounds are kept to a low intensity then they are less likely to annoy other computer-users in the vicinity as they will not be able to hear them. Sounds giving constant feedback should be kept to just above threshold. Sounds to provide notification of important events should be loud enough to capture the user's attention but still quiet so that others are not forced to hear them. This is possible because, as mentioned above, most people sit very close to their machines. As described in the previous chapter, volume is not the only way of making a sound demanding. Changing the rhythm of a sound can be as effective because of the auditory system's ability to detect changing stimuli.

Headphones could be used so that sounds are only heard by the primary user of the system. This is not a good solution as many users want to be able to hear the other sounds in their environment and communicate with colleagues. Bregman [29] suggests a futuristic ear-piece that could perhaps present sounds only to the primary user and allow everyday sounds in. Svean [165] describes such a system: The Active Ear Plug (AEP) which is currently under development. When such equipment becomes generally available many of the problems of annoyance due to intensity could be overcome.

In the experiments described in this chapter the annoyance due to sound feedback was measured to find out if it was a problem. These experiments were on individual subjects so would only give information about annoyance to the primary user and not to users who overheard the sounds. There has been no other investigation of the potential annoyance due to auditory interfaces so this research is a first step.

7.3 EXPERIMENTAL TESTING FRAMEWORK

In order to test the sonically-enhanced widgets an experimental testing framework was created. This would allow the testing of all the widgets in a simple and consistent manner. The same types of measures and designs would be used for each one. The experiments would be short and simple to evaluate. This would allow several widgets to be investigated quickly to see if sound would improve usability.

The framework was designed around the measures suggested by Bevan & Macleod [19]. They say that any system for testing usability should measure *effectiveness* (the accuracy with which goals are achieved), *efficiency* (the relationship between efficiency and expenditure of resources) and *satisfaction* (the perceived usability of a system). The measures used here (see below) assessed effectiveness by examining error rates, efficiency by time or amount completed, and satisfaction by measuring workload and overall preference. The testing framework should therefore give reliable and complete information about the effects of sonic-enhancement on the usability of the new widgets.

Subjects	Condition 1		Condition 2	
Six Subjects	Sonically-enhanced Widget Train & Test	Workload Test	Visual Widget Train & Test	Workload Test
Six Subjects	Visual Widget Train & Test		Sonically-enhanced Widget Train & Test	

***Table 7.1**: Format of the experiments.*

7.3.1 Design

A two-condition, within-subjects design was used to test all of the widgets. This allowed each subject to be used twice and their results compared. In one of the conditions the standard graphical widget was tested, in the other condition the sonically-enhanced widget. The order of presentation was counterbalanced to avoid learning effects: Six subjects did the auditory condition then the visual, six did the visual then the auditory. Table 7.1 shows the format of the experiment. Before each condition the subjects were trained on the interface they were about to use. After the test of each condition subjects were presented with workload charts which they had to fill-in (this is described in detail below). An example of the workload charts are shown in Appendix C Table 1. Instructions were read from prepared scripts.

All the sounds used were played on a Roland D110 multi-timbral sound synthesiser. The sounds were controlled by an Apple Macintosh via MIDI through a Yamaha DMP 11 digital mixer and presented to subjects by loudspeakers.

7.3.2 Subjects

Each experiment used twelve expert subjects: Postgraduate and undergraduate students from the Department of Computer Science at York. All had more than three years experience of using graphical interfaces. Experts were used because the types of problems being investigated were *action slips* (see Chapter 6 or [124]). As described in the previous chapter, this type of error commonly occurs with expert users. They perform many interface tasks automatically without monitoring the feedback from the interaction because the task is well known. Novice subjects might suffer from similar problems but these would be for different reasons, for example hand-eye coordination or mouse control difficulties.

7.3.3 Tasks

Lee [105] has suggested that short experiments of the type that the testing framework proposes are not good for investigating action slips. He says that longer term experiments are needed so that subjects have time to become familiar with the experimental task they must perform. They must have time to develop skill with the task so that only low-level monitoring of feedback is performed.

As part of this research several widgets had to be tested to find out if the event, status and mode analysis technique was effective. Due to time constraints the experiments had to be short. In order to keep them short but also to avoid the problem Lee describes, the tasks that the subjects had to perform were kept very simple. A simple task was needed so that subjects could learn it quickly and rapidly reach a level of automaticity where only low-level cognitive monitoring would be performed. This would then be similar to the normal, everyday use of the interface.

7.3.4 Measures

In order to get a full range of quantitative and qualitative results time, error rates and workload measures were used as part of the framework. Time and error rate reductions would show quantitative improvements and workload differences would show qualitative differences.

Workload testing

Wright & Monk [184] have argued that quantitative measures of usability such as error rates and performance times do not provide a complete picture of the usability problems of an interface. They argue that users may perform tasks well and quickly and yet find the system frustrating to use and that it requires more effort to complete a task than they would expect. This dissociation between behavioural measures and subjective experience has also been addressed in studies of workload. Hart and Wickens ([92], p 258) for example, define workload "...as the effort invested by the human operator into task performance; workload arises from the interaction between a particular and task and the performer". Their basic assumption is that cognitive resources are required for a task and there is a finite amount of these. As a task becomes more difficult, the same level of performance can be achieved but only by the investment of more resources. A workload test was used as part of the usability evaluation of the sonically-enhanced widgets discussed in this paper. It provided a more rounded and sensitive view of usability than just time and error analysis alone.

The NASA Human Performance Research Group [122] analysed workload into six different factors: Mental demand, physical demand, time pressure, effort expended, performance level achieved and frustration experienced (a brief description of each of the factors is given in Appendix C Table 2). NASA have developed a measurement tool, the NASA-Task Load Index (TLX) [91, 122] for estimating these subjective factors. As Bevan & Macleod ([19], p 143) say:

> “The Task Load Index (TLX) is a multi-dimensional rating procedure that provides an overall workload score based on a weighted average of ratings on six subscales.”

Three of the subscales relate to the demands imposed on subjects in terms of:

- the amount of mental and perceptual activity required by the task
- the amount of physical activity required
- the time pressure felt

A further three subscales relate to the interaction of an individual with the task:

- the individual’s perception of the degree of success
- the degree of effort an individual invested
- the amount of insecurity, discouragement, irritation and stress felt

TLX has been tested in a wide variety of experimental tasks that range from flight simulators to laboratory tests of problem solving. Workload measures are little used in the field of interface evaluation yet the six factors identified in TLX would appear to bear directly on usability. Thus it would seem valuable to incorporate an estimate of workload into an evaluation of the sonically-enhanced widgets.

When performing a TLX workload analysis subjects rate each of the six factors on rating scales. Normal TLX analyses then use paired-comparisons between each of the factors to derive weights for a user’s subjective feeling of importance for a particular factor. The individual factor scores are multiplied by the weights then an average taken to give the overall workload score. The application of TLX is therefore a two-pass procedure. To speed up the application of TLX, Byers, Bittner & Hill [40] proposed that the weightings were not needed; instead an average of the factor scores could be used to give the overall workload. They carried out a comparison of traditional TLX (with weightings) and ‘raw’ TLX (just the average). Their results showed no significant differences. They conclude (p 484):

> "RTLX [raw TLX] is attractive for use because of its simplicity and essential equivalence with TLX. Because of its simplicity, we believe it has substantially greater potential in industrial and research settings than its predecessors. RTLX is recommended for use as a tool for multidimensional assessment of operator workload."

The workload analyses performed in this chapter used the raw TLX method. The basic six factors were used as described but a seventh factor was added: Annoyance. This is often cited as a reason for not using sound in an interface as it is argued that continued presentation of sound would be an annoyance for the user. So, by adding this as a specific factor in the usability assessment it would be possible to find out if subjects felt that sonic feedback was an annoyance.

In addition to these seven factors subjects were also asked to indicate, overall, which of the two interfaces they felt made the task easiest. This would give an overall subjective preference measure. An example of the workload charts used in the experiments is shown in Appendix C Table 1. Appendix C Table 2 shows the workload descriptions given to the subjects when they were filling-in their workload charts. These were taken directly from the NASA TLX guide [122]. At the beginning of each experimental investigation, before the first condition, an overall description of workload was given to the subjects so that they understood what workload was and how to fill in the charts shown in Appendix C Table 1. The final workload bar, 'overall preference', was filled in after both of the conditions had been completed.

7.4 TIMBRE USED IN THE SONICALLY-ENHANCED WIDGETS

In the scrollbar and button widgets investigated below an electric organ timbre was used for all of the sounds. The reason for this was that if an interface was sonified each application would have its own timbre. The experiments in Chapters 4 and 5 showed that timbre was a major discriminating feature of earcons. If each application was given its own timbre users would learn to associate it to the application and recognise that, for example, electric organ meant word processor or acoustic piano a drawing package. Within an individual application all the sonically-enhanced widgets would use the base timbre and then modify it with different rhythms, pitches, etc. as necessary creating a hierarchy of earcons for each application. In different applications widgets would have different timbres but would otherwise sound the same. Therefore, in the first two experiments described, the same timbre was used to show that all the auditory feedback from an application could be produced with one timbre. In the window experiment different timbres were used for the two windows. This simulated the situation where each application had its own timbre. In this case, each window had its own timbre and spatial location. The earcons within each window were the same but were based on the window timbre. This was done to investigate if the overall sonification described was possible.

The following sections describe the use of the experimental testing framework to investigate three sonically-enhanced widgets. These use earcons for audio output and the event, status and mode analyses from the previous chapter indicate where to add the sound.

7.5 THE EVALUATION OF A SONICALLY-ENHANCED SCROLLBAR

The first widget investigated was the sonically-enhanced scrollbar. In Chapter 6 an ESM analysis of the problems due to hidden information in scrollbars was performed. The experiment described here attempts to test this analysis to discover if the solutions suggested would improve usability. Three problems with scrollbars were discussed: Dragging the thumb wheel out of the 'hot spot', losing one's sense of position in a document and 'kangarooing' with the thumb wheel. The latter two problems are experimentally tested here. The experiment was based on the framework described above. In one half of the test subjects were given a standard visual scrollbar and in the other subjects were given a sonically-enhanced one (see Table 7.1). This experiment was first published in [35].

7.5.1 Tasks

Subjects were given two types of task. The first, which will be called the *Search Tasks*, involved the subjects visually searching through a file of data to find significant features. These features were such things as whole line of 'a's together. When the target was found the subjects had to say which page the target occurred on. The other tasks, which will be called the *Navigate Tasks*, involved subjects being given instructions to go to a specific point on a specific page and read the data that was there. For example, subjects were asked to go to page seven and read the first six characters of the first line. Along with these absolute navigation tasks relative tasks were also given. For example, subjects were asked to go up four pages from the current page and read the first six characters of the last line. These two types of tasks cover some of the main ways users interact with scrollbars. They might be searching through a document to find something or they might be looking for a specific page to find the data they want. The data were described to the subjects as 'experimental results data'. The rationale given to the subjects for the tasks was that they were searching through the data to find significant features for analysis.

7.5.2 Sounds used

The earcons were designed using the guidelines from Chapters 4 and 5. Two sounds were added to create the sonically-enhanced scrollbar. These were described in Chapter 6. Figure 7.1 shows the hierarchy of earcons used in the experiment.

Thumb reaching target event

A fixed tone of duration 9/60 sec. (0.15 seconds) was used to indicate a window scroll event. When the subject scrolled towards the bottom of the document a low-pitched note, C_4 (130Hz), was played. When scrolling up a high-pitched note C_0 (2093Hz) was played. The intensity of the sounds was raised over the background, continuous position in document sound. This, combined with the change in pitch, made the sound demanding. Using the combination of pitch and intensity meant that the earcons did not have to be as loud as if only intensity had been used. This would therefore be less annoying for others nearby. If the subject was scrolling downwards towards a target location he/she would hear the low-pitched sound. If kangarooing occurred then the subject would hear a demanding high-pitched tone when not expected.

Page scrolling/position in document indication

A low intensity continuous tone gave status information about the current page. To indicate a page boundary event the background tone was increased in volume for two beeps of 9/60 sec. each to demand the listener's attention. It then decreased again to just above threshold level so that it could be habituated. The notes played when scrolling towards the bottom of the document decreased in pitch from B_1 (1975Hz) to C_4 (130Hz) when a page boundary was crossed. The notes played cycled through the scale of C major. So, for example, when scrolling down from the top of the document, the first note played would be B_1 (1975Hz), then A_1, G_1, F_1, E_1, D_1 and on to C_2 of the octave below. The reverse occurred when scrolling up from the bottom of the document.

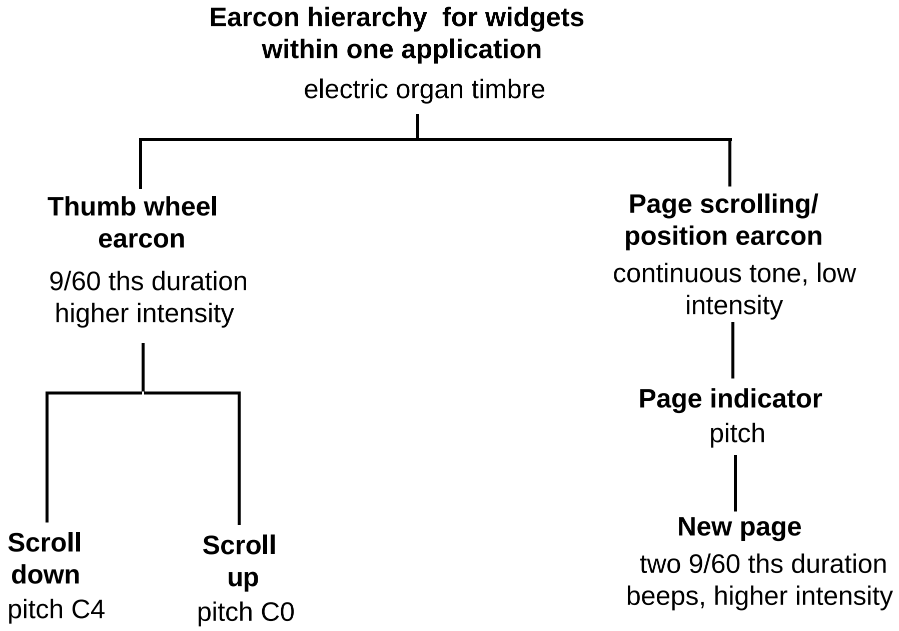

Earcon hierarchy for widgets within one application
electric organ timbre
Thumb wheel earcon
9/60 ths duration higher intensity
Page scrolling/ position earcon
continuous tone, low intensity
Page indicator
pitch
Scroll down
pitch C4
Scroll up
pitch C0
New page
two 9/60 ths duration beeps, higher intensity

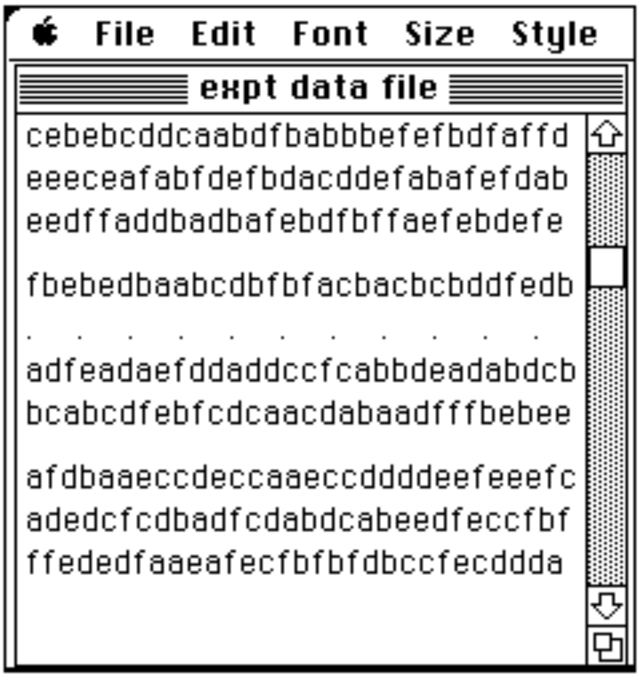
Figure 7.1: *The hierarchy of earcons used in the scrollbar experiment.*

When the scrollbar was clicked the thumb sound was played first followed by the page boundary sound after a 9/60 sec. delay (if a page boundary had been crossed).

7.5.3 Experimental design and procedure

The experiment was in two halves and was based on the design from the testing framework described above (see Table 7.1). A simple document browser was created on an Apple Macintosh, based around TinyEdit, an example program supplied by Symantec Corporation with the Think Pascal compiler (see Figure 7.2). This browser allowed subjects to navigate around a document using a scrollbar and indicated page boundaries with a dotted line, in a similar way to many wordprocessors. The scrollbar used in the browser only allowed clicking in the grey scroll area above or below the thumb wheel to scroll by a window of data either way. The subjects could not drag the thumb wheel or scroll by lines using the arrows. This was done because the experiment was to investigate kangaroo problems and these only occur when clicking in the scroll area.

The data files used in the browser were made up of groups of three lines of 30 randomly generated ‘a’ to ‘f’ characters separated by a blank line. The test files had twelve pages of data where pages were 50 lines long and windows 33 lines long. Therefore, scrolling by a window did not necessarily mean that a new page boundary would be reached each time. The data was displayed in 12 point Geneva font. Figure 7.3 shows an example of the data in a test file. There was a different test file for training, the auditory condition and the visual condition so that subjects would not be able to learn their way around the



```
cebebcddcaabdfbabbbefefbdfaffd
eeeceafabfdefbdacddefabafefdab
eedffaddbadbafebdfbffaefebdefe

fbebedbaabcdbfbfacbacbcbddfedb
.    .    .    .    .    .    .    .    .    .    .
adfeadaefddaddccfcabbdeadabdcb
bcabcdfebfcdcaacdabaadfffbebee

afdbaaeccdeccaaeccddddeefeeefc
adedcfcdbadfcdabdcabeedfeccfbf
ffededfaaeafecfbfbfdbccfecddda
```

Figure 7.2: *The browser program (reduced in size) used in the scrollbar experiment.*

```
dceedfbddeddbbcbbcadcdfbcebdee
bafadebddbcdbafeafbaebdccabcce
abdeededeaffbddcdfebaeddfbcebc

dceedfbddeddbbcbbcadcdfbcebdee
bafadebddbcdbafeafbaebdccabcce
abdeededeaffbddcdfebaeddfbcebc

aaaaeacffccfcacecffcfbedbdeceb
bedbcaeabfecafafafdbfdeecbfbba
eacebfdadfbfcfcdeecbacbaadcfef
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

bbfefceeaddffedabdceeeadfeafdf
ccdefceeaddffedabdcfdeadfeaaea
eabafdbaadaeaabdfcdbeedfeccfbf

aaaaaaaaaaaaaaaaaaaaaaaaaaaaa
ccbaeacffccfcacecffcfbedbdeceb
bedbcaeabfecafafafdbfdeecbfbba

baccdfecabbaaddfeeaacbcdeecfbe
ebfcdacbcaedcecdaefaefaebbcbed
adedecbbbaffdacaaefdbfcdfaeadc
```

Figure 7.3: *A sample of the data appearing in the test files for the scrollbar experiment. The dotted line shows a page boundary and the line of ‘a’s shows one of the targets.*

data files. If they had learned it then the second condition of the experiment would have shown a learning effect.

Visual condition

In the visual condition, subjects used an ordinary Macintosh scrollbar. Training was given in both types of task before the main test was started. The experimental procedure was described and then sample Search and Navigate tasks were undertaken using a training data file. In the main test subjects were given a task by the experimenter and when they were ready to start they pressed ⌘Y to start a timer. When they had completed the task they pressed ⌘Y again, the timer was turned off and the time recorded. Errors were recorded by the experimenter. A subject was given the search task questions first and then the navigate ones. When the subject found a target in the search task he/she gave the page number to the experimenter. If it was incorrect then the correct page number was given. This was necessary so that the subject started from the correct page when searching for the next target. For the navigate tasks, the subject gave the required six characters. If these were incorrect then the subject had to search until he/she found the correct page.

Auditory condition

The sonically-enhanced scrollbar described above was used. In the initial training of subjects for this condition the feedback provided by the scrollbar was described in detail. The training and testing then proceeded as described above for the visual condition.

7.5.4 Experimental hypotheses

The hypotheses were based around the predictions of the ESM analysis technique described above. The model suggested that recovery from kangaroo errors would be quicker as users receive demanding feedback indicating when errors happen rather than noticing later on when they were not where they expected to be. Subjects can therefore correct the errors more quickly.

Subjects should better be able to maintain their sense of position in the document with more page feedback and therefore give fewer wrong page answers. If subjects lost their sense of position the time cost was high. For example, they would have to go back to the top of the data file and work out their position from there. This would take much time. Therefore, if they did not lose their sense of position, time to complete tasks should be reduced. The demanding audio feedback should make it easier for subjects to perceive page boundaries and so make fewer wrong page errors.

The workload felt by subjects should be reduced as the extra feedback provided information that they needed. Subjects would have to expend less effort recovering from errors and remembering whereabouts in the document they were. Physical demand and time pressure would be unaffected as they were unchanged across conditions. There would be no increased frustration or annoyance due to the addition of sound as the auditory feedback provided information that the subjects needed.

7.5.5 Results

TLX results

Figure 7.4 shows the average scores for each of the workload categories plus the two extra ones. Each was marked in the range 0-20. Paired T-tests were carried out on the auditory versus visual conditions for each of the workload categories. Average total raw workload (based on the six standard TLX factors) for the auditory condition was 9.61 and for the visual 10.25. These scores were not significantly different (T(5)=1.35, p=0.234). Mental demand was assigned the highest mark of all the workload scores indicating that the experimental task itself was difficult. It showed a significant decrease in the auditory condition over the visual (T(11)=3.23, p=0.008). Nine of the twelve subjects rated the auditory condition lower in effort than the visual but this failed to reach significance (T(11)=1.83, p=0.09). There were no significant differences in any of the other workload categories.

The annoyance for the auditory condition was not significantly different to the visual condition (T(11)=0.516, p=0.615). Five subjects rated the auditory condition more annoying than the visual and three rated the visual as more annoying than the auditory. There was a difference in terms of overall preference. In this case, the subjects were asked to rate which scrollbar made the task the easiest. Here the auditory scrollbar was significantly better than the visual one (T(11)=2.55, p=0.02). The raw data for the workload analysis are shown in Table 3 of Appendix C.

Timing and error results

Along with workload tests, conventional measures of time and error rates were taken. The raw data are shown in Appendix C Table 4. Figure 7.5 shows the total times taken by each of the subjects in the two conditions for the search tasks. Nine of the twelve

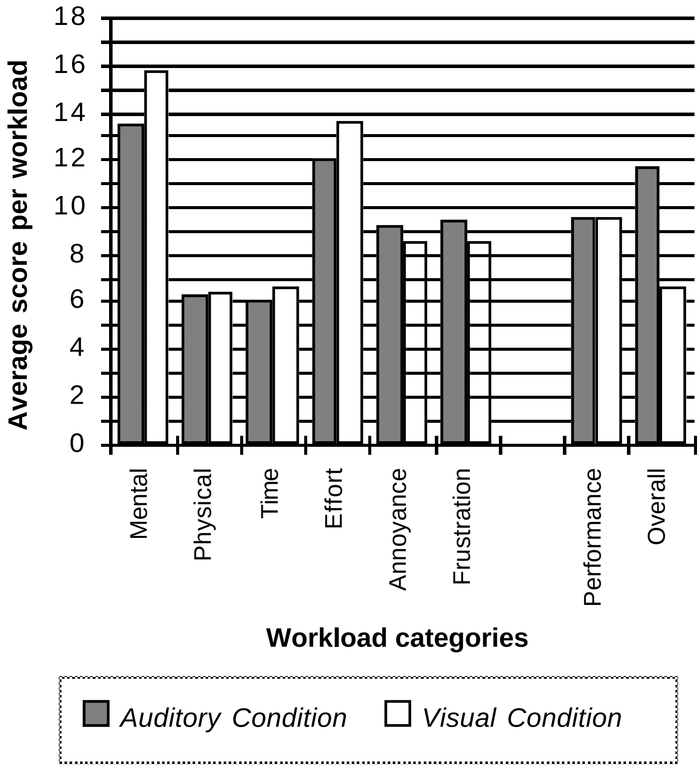


Figure 7.4: *Average TLX workload scores for the auditory and visual conditions of the scrollbar experiment. In the first six categories higher scores mean higher workload. The final two categories, performance and overall, are separated because higher scores mean less workload.*

subjects performed faster in the auditory condition but there was no significant difference in time scores at the 95% level (T(11)=1.846, p=0.09). However, an F-test between the auditory and visual conditions across subjects showed a significant reduction in the variance in the auditory condition (F(11,11)=3.98, p=0.05).

To find out if any underlying differences were hidden in the overall timing data a more detailed analysis was undertaken. The average time taken to answer a question where errors occurred was calculated for each question in both conditions of the search tasks. Both types of errors were included in this analysis because of the small numbers of kangaroo errors. There were no significant differences between the conditions in time taken to answer questions with errors (T(2)=1.24, p=0.33) or to answer questions where there were no errors (T(2)=1.39, p=0.29).

Two kinds of errors were recorded: Kangaroo errors and wrong-page errors (see Table 7.2). There were no significant differences in either of the error rates between the two conditions.

Figure 7.5 shows the total times for the two conditions in the navigate tasks. In these tasks there was a significant difference between the times taken. A paired T-test showed the auditory condition was significantly faster than the visual (T(11)=2.29, p=0.04). As before, there was also a significant reduction in the variance in the auditory condition (F(11,11)=5.43, p=0.05). To find whether the decrease in time taken for the auditory condition was due to faster recovery from errors, a more detailed analysis was undertaken. Recovery from errors was significantly faster in the auditory than in the visual condition (T(9)=2.61, p=0.02). The average time taken to answer questions with no errors was also calculated. A paired T-test showed that the auditory condition was again significantly faster than the visual (T(9)=4.18, p=0.002).

Once again, there were no significant differences in the error rates between the two conditions. However, there was a reduction in both categories of error in this task. For example, the number of wrong-page errors fell from 51 to 40 in the auditory condition but this failed to reach significance (see Table 7.2).

Tasks/ Conditions	Search		Navigate	
	Wrong page	Kangaroos	Wrong page	Kangaroos
Auditory	13	5	40	4
Visual	11	3	51	8

***Table 7.2**: Totals of wrong page and kangaroo errors in both conditions of the scrollbar experiment.*

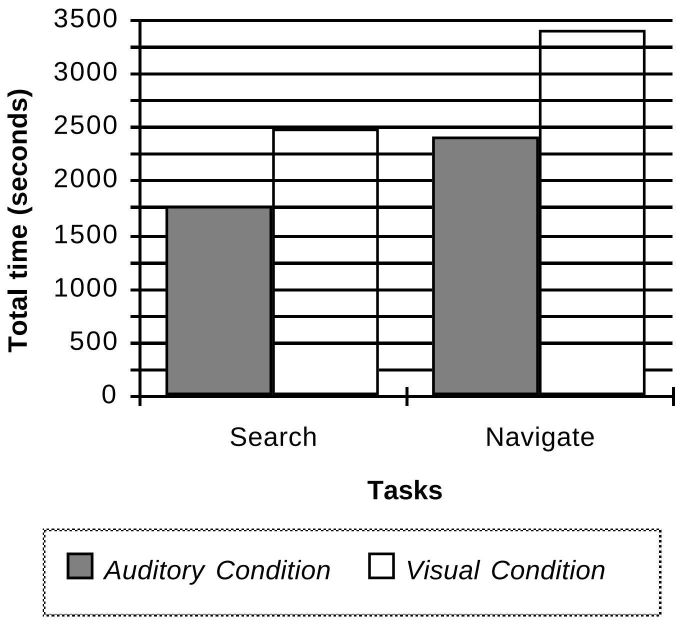


Figure 7.5: *Total times for the search and navigate tasks in the scrollbar experiment.*

7.5.6 Discussion

The workload results indicated that the auditory scrollbar reduced the workload of the task. Mental demand (which dealt with how much mental and perceptual activity was required) was significantly reduced. This could be due to it being easier for subjects to hear page boundaries than it was to see them as the feedback was more demanding. Subjects also got more feedback about kangaroo errors so making it less effort to recover from them. This confirmed the hypothesis that extra auditory feedback would lower workload. Although subjects felt their performance was no better in the auditory condition than in the visual, they had an overall preference for the auditory scrollbar because it lowered mental demand and there was some decrease in effort expended. These factors indicated that an auditory enhanced scrollbar would be an effective addition to an interface and could lower the workload therefore freeing-up cognitive resources for other tasks.

There was no significant difference in the annoyance or frustration felt by subjects in the auditory condition. This indicated that auditory feedback, and especially constant auditory feedback, was not necessarily annoying when used at the interface. This confirmed the hypothesis that auditory feedback would not be annoying if it provided useful information to the user.

The significant reduction in time for the auditory condition in the navigate tasks indicated that the auditory enhanced scrollbar improved performance. This is again evidence to suggest that auditory scrollbars are an effective extension to standard visual ones. The times for the search tasks were not significantly different. This may have been due to the nature of the task. A subject was required to visually search through the data file to find a target. The advantage conferred by sound may have been lost in the overall time to do the visual searching. This visual searching took up a large proportion of the time for this task and the position awareness within the document was bound up in this. The advantages due to sound were small and therefore lost in the large times for visual searching. In the navigate tasks, where the subjects had to find a specific page, searching was based on page boundaries so there was a better comparison between the auditory and visual conditions.

There were no significant differences between conditions in the time taken to recover from errors in the search tasks. As described previously, the time to do the searching might have been the problem here. In the navigate tasks the auditory group was significantly faster overall. When this result was investigated in more detail the auditory condition was found to be significantly faster at recovering from errors than the visual condition. The auditory condition also performed better when there were no errors. It seems that the sounds helped increase general performance with the scrollbar, perhaps in a similar way to the general raising of mode awareness in Monk’s [117] experiment described in the previous chapter.

A problem with the error analysis was that the frequency of kangaroo errors was too low to be a good measure. For example, in the search tasks there was less than one error per subject in each of the conditions. It turned out to be very difficult to generate many kangaroo type errors. It could be that as the subjects were experienced scrollbar users they had developed strategies for avoiding kangarooing in their everyday work which they used in the experiment. However, two subjects said that the sounds did help them identify when a kangaroo error had taken place. The problems generating kangaroo errors meant that it was difficult to test the hypothesis that recovery from such errors would be quicker. They had to be combined with wrong page errors and an overall analysis performed.

There were no differences between the conditions in the number of wrong-page errors. It may have been that subjects counted the page boundaries whether they saw them or heard them, but it just took longer when they had to do it visually. This may have been one of the reasons for improved performance in the navigate tasks for the auditory condition. Further investigation of errors is therefore necessary.

It is noteworthy that there were significant differences between the auditory and visual conditions in terms of variance on both tasks. Eight of the twelve subjects showed less variability in the auditory condition. However, a Sign test between conditions across subjects failed to reach significance. There is an indication that the variability has been reduced and further experiments would be needed to investigate this.

Justification of the event, status and mode analysis method

The sonically-enhanced scrollbar was designed to overcome the problems identified by the ESM analysis. Do the experimental results justify the technique? The addition of sound produced a significant improvement in performance in one of the tasks and a decrease in the overall variability in both tasks. The mental workload required to perform the task was significantly less when sound was used and overall preference was for the auditory scrollbar. All these results indicate that the addition of sound was successful and the ESM model was proven to be effective. One area which needs further investigation is error rates. The model predicts that the number of wrong-page errors should be lower but the results failed to demonstrate this because not enough errors were generated in the experiment.

7.5.7 Future work

In addition to giving information about page boundaries other events could be indicated when scrolling. In a programming editor, for example, events such as when a new procedure or function was reached could be displayed in sound. The scrollbar only allowed sounds for 21 pages due to its use of pitch. This could be extended by using different rhythms or intensities along with pitch so that bigger documents could be dealt with. The continuous tone of the page indicator could be made to fade into the background of consciousness by lowering its volume if the user stayed on the same page for a period of time. Currently the sound remains at the same volume but lowering it would help habituation.

7.5.8 Summary

A sonically-enhanced scrollbar was tested and found to significantly reduce performance time and error recovery on certain tasks. It also significantly reduced the mental workload and was rated with a significantly higher preference score than a standard visual scrollbar. There was also no increased annoyance due to sound. This indicated that the integration of auditory feedback into graphical widgets was likely to provide more usable interfaces. The use of workload tests is also shown to be a useful way of measuring usability. The results from the experiment suggest that the event, status and mode analysis technique is effective in identifying areas in the interface where problems occur. Until now there was no structured approach to adding sound to an interface, it was done in an *ad hoc* way by individual designers. The results of the work described here show that widgets that combine both auditory and visual feedback are more effective as they make use of the natural way that humans deal with information in everyday life.

7.6 THE EVALUATION OF SONICALLY-ENHANCED BUTTONS

In Chapter 6 some of the problems of graphical screen buttons were investigated by ESM analysis. The main problem was the user slipping-off the screen button before the mouse button was released. An experiment was performed, based on the framework described above, to test the solution proposed by the analysis to see if it improved usability.

In Chapter 6 an analysis was performed of button slip-off errors. A solution to one other button problem will be tested in this chapter. In some situations, for example when first encountering a new interface, it can be difficult to know what parts of the display are buttons and what are not because there may be many icons on the screen. The user may spend time trying to find buttons that can be pressed. Examples of this type of problem may be clicking on the image of a figure on the screen to give information. This image will not look like a standard button that the user is familiar with. Another problem associated with this is that some buttons can be very small and it can be hard to position the mouse on them. This is especially true on personal digital assistants (PDA's), or other portable computers with small screens, where hardware limitations (lack of screen size) cause the targets to be small and hard to hit. This problem of hardware limitation causing information to be hidden (or made difficult to get at) was one of the problems described in the previous chapter.

To solve this new problem audio feedback can also be used. The problem occurs when the mouse is moved over the screen button. As shown in status feedback 1 of the real button in Table 6.4 of the previous chapter, status feedback from the button is action-independent, avoidable, sustained and static. One reason the feedback is avoidable is because the user might not be able to tell what is and what is not a button. An action-independent sound could not be used because then sounds from all of the buttons would play all of the time. As described in the previous chapter, action-independent feedback can be converted into action-dependent. Sound could be played whilst the mouse was over the screen button. The sound could be sustained until the mouse was moved off the screen button. This might only be for a short period of time (the same as the highlight of the screen button described in Table 6.4). The sound would be static; it would only indicate that the mouse was over the screen button. The sound would start on the event of moving the mouse on to the screen button. It would be sustained as status information until the mouse was moved off the screen button. The sound would be demanding because when the mouse was moved over the screen button the sound would start and this change in stimulus would be noticeable. Extra graphical feedback could be used to give this information. The outline of the button could be highlighted, for example, but again this would increase the complexity of the visual display.

An experiment was designed to test sonically-enhanced buttons, based on the framework described above, to see if they would be more usable than conventional graphical ones. An initial experimental design was piloted on five subjects but failed to cause any mis-hit errors (Dix & Brewster [61]). The experiment was redesigned paying greater attention to the analysis of why slip-off errors occurred. This is the experiment reported here.

7.6.1 Task

Figure 7.6 shows a screen shot of the interface to the task. Subjects were required to enter five digit codes (Figure 7.7 shows some sample codes from the experiment). The codes were randomly generated and a different set of codes was used in each condition of the experiment. These codes were displayed in the ‘Code to type’ field. To start the experiment the ‘Start’ button was pressed. This put the first code into the code to type field. The subjects then had to enter the code using the mouse and on-screen keypad. To do this they had to press a number and then hit the ‘OK’ button to accept it, then press the next number and hit the ‘OK’ button and so on. The numbers entered appeared in the ‘Code’ field above the keypad. A delete key was provided so that the subjects could correct mis-typed codes. When the code had been typed the ‘Next’ button was used to display the next code. This maximised the number of button presses and mouse movements the subject had to make. In the visual condition the buttons acted like normal Macintosh buttons. They went black to indicate the mouse had been pressed over a button and went back to white when the mouse was released or moved off the button. In the auditory condition there was no visual highlighting (the buttons stayed white). The buttons made the sounds described below.

7.6.2 Sounds used

The sounds used were based around the earcon guidelines put forward in Chapters 4 and 5. A hierarchy of earcons was created (see Figure 7.8). A base sound was created for when the mouse was moved over a screen button. This was a continuous tone at C_4 (130Hz). The volume of this was kept to just above the threshold level. This indicated to the user when the mouse was over a screen button. The sound captured the subject’s attention when it came on, even though it was at a low intensity, because it was a new stimulus. When the mouse button was pressed down over a screen button a higher pitched sound at C_3 (261Hz) was played at a higher volume. This continued for as long as the mouse button was down and the mouse was over the screen button. If the mouse was moved off the screen button the sound stopped. If the mouse was released over the screen button then a success sound was played. This consisted of two notes, played consecutively, at C_1 (1046Hz) each with a duration of 2/60 sec. This success sound had to be kept short so that the user did not get confused as to which button the feedback was coming from: The audio feedback had to be able to keep pace with interactions taking place. To make sure that the number of sounds was kept to a minimum and speed maximised, if the user quickly clicked the mouse over the screen button the mouse down sound was not played; only the successful click sound was played. The mouse button down and success sounds differentiated a successful and unsuccessful mouse click.

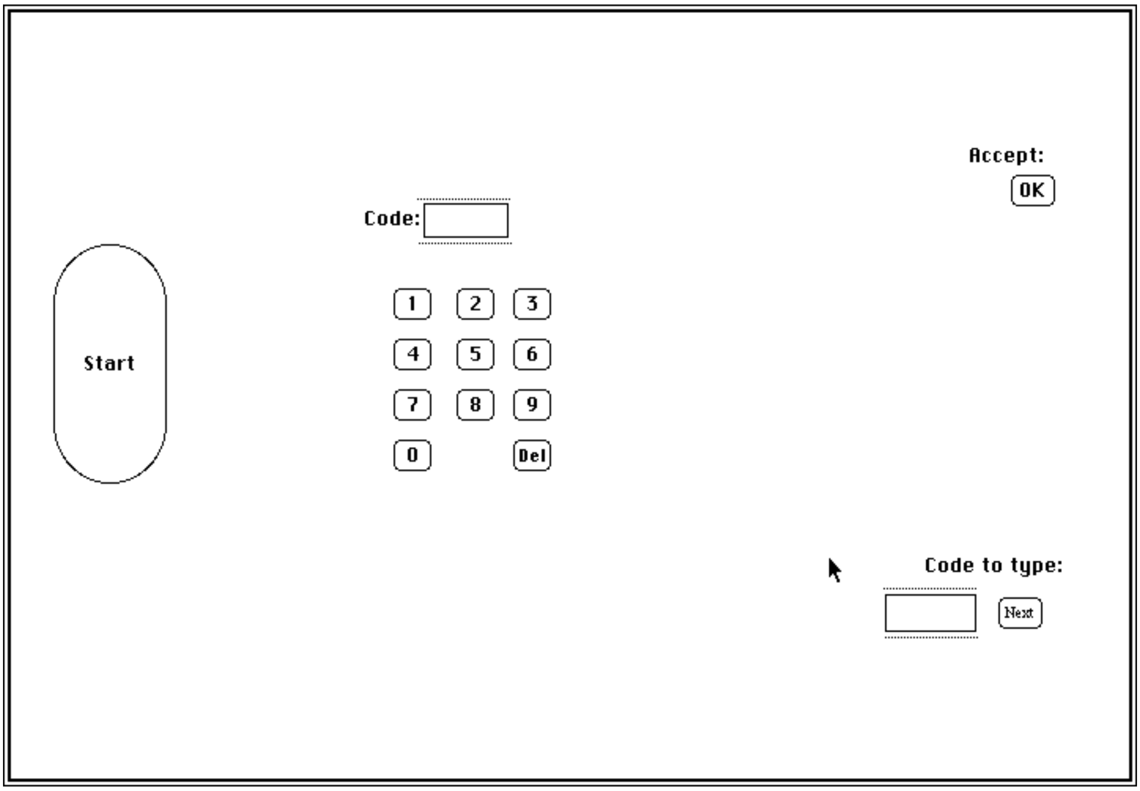


Figure 7.6: *The button testing program.*

40822
26100
70585
40710
65307
95419
41236
87619
78727

Figure 7.7: *Sample codes used in the buttons experiment.*

The mouse down and success sounds used a combination of pitch and intensity to make them demanding. As in the scrollbar sounds, described above, this meant that a lower intensity could be used making the sounds less annoying for others nearby.

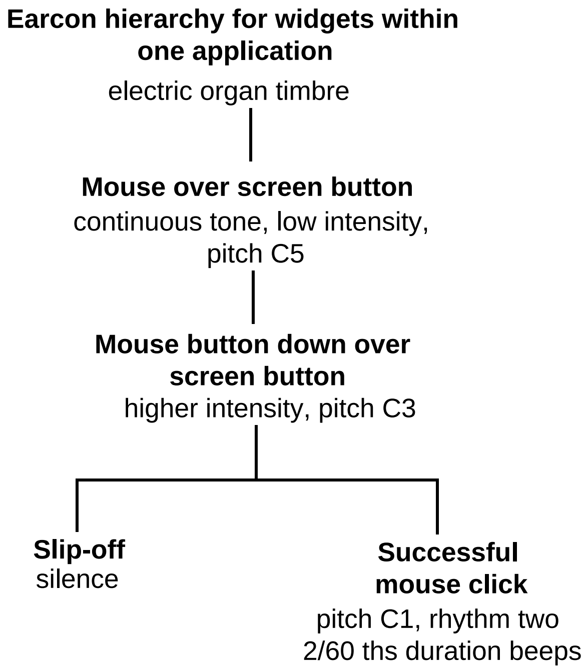

Earcon hierarchy for widgets within one application
electric organ timbre
Mouse over screen button
continuous tone, low intensity, pitch C5
Mouse button down over screen button
higher intensity, pitch C3
Slip-off
silence
Successful mouse click
pitch C1, rhythm two 2/60 ths duration beeps

Figure 7.8: *The hierarchy of earcons used in the buttons experiment.*

The mouse button down over the screen button earcon could be seen as redundant because the mouse over sound gave effectively the same feedback. It told the listener when the mouse was over a button and when it had moved off. The mouse down sound was there to mimic the mouse button down over screen button feedback from the visual condition. Feedback is needed to indicate to the user when the mouse has been pressed over the screen button. It did, however, have another purpose. If there was no mouse button down over screen button feedback then the mouse over sound would have had to be more demanding, so that the user knew he/she was over the button. The mouse over sound was just above hearing threshold and so could be avoidable. This was very useful because the user would not want a demanding sound played every time the mouse moved over a screen button; it would become annoying. Thus, having the extra sound allowed the mouse over sound to be avoidable.

Having a sound that came on when the mouse was over target indicated to users that they had reached a target. The sound could be turned off if, for example, a button was not available at a particular time (similar to greying-out). The user would then know that it was not available.

7.6.3 Experimental design and procedure

The experiment was in two halves and was based on the design from the testing framework described above (see Table 7.1). Training was given before each of the conditions so that subjects could get used to the mouse and the method of entering data.

Each condition lasted 15 minutes and the subjects had to type in as many codes as they could.

Three main types of data were recorded. The total number of codes typed by a subject in the fifteen minutes was stored. Two different types of errors were recorded. The first was the number of clicks which totally missed a target, which were called *background clicks*. Here, neither the mouse-downs or mouse-ups were over a button, i.e. the subject missed the buttons completely. The other type of error was a slip-off. Here the subject clicked on the screen button but moved the mouse off before releasing the mouse button. For each of the codes typed a trace was recorded of all the mouse button up and downs with time stamps. This trace could then be analysed to find the slip-off errors that had occurred. All the data on errors and codes typed was recorded by the computer into the trace.

7.6.4 Experimental hypotheses

The hypotheses were based around the predictions of the ESM analysis technique described previously. The extra feedback provided by the sounds should make it easier for subjects to recover from errors. They would notice that the errors have occurred earlier than in the visual condition. More codes should be typed in the fifteen minutes because less time will be spent on error recovery. There should be fewer buttons misses because the feedback indicated when the mouse was over the button.

The workload felt by subjects should be reduced as the extra feedback would provide information that the subjects need. Subjects should have to expend less effort recovering from errors. Physical demand and time pressure should be unaffected as they are unchanged across conditions. There should be no increased frustration or annoyance due to the addition of sound as the auditory feedback will provide information that the subjects need.

7.6.5 Results

TLX results

Figure 7.9 shows the average workload score for each category. Average total raw workload (based on the six standard TLX factors) for the auditory condition was 8.6 and for the visual 9.04. These scores were not significantly different (T(5)=1.75, p=0.147). An analysis of the individual scores on the NASA TLX tests showed that none of the scores were significantly different between the two conditions. For example, there was no significant difference between the mental workload scores across the conditions (T(11)=1.46, p=0.169). This indicates that the auditory buttons did not significantly lower the workload of the task. However, they were rated very

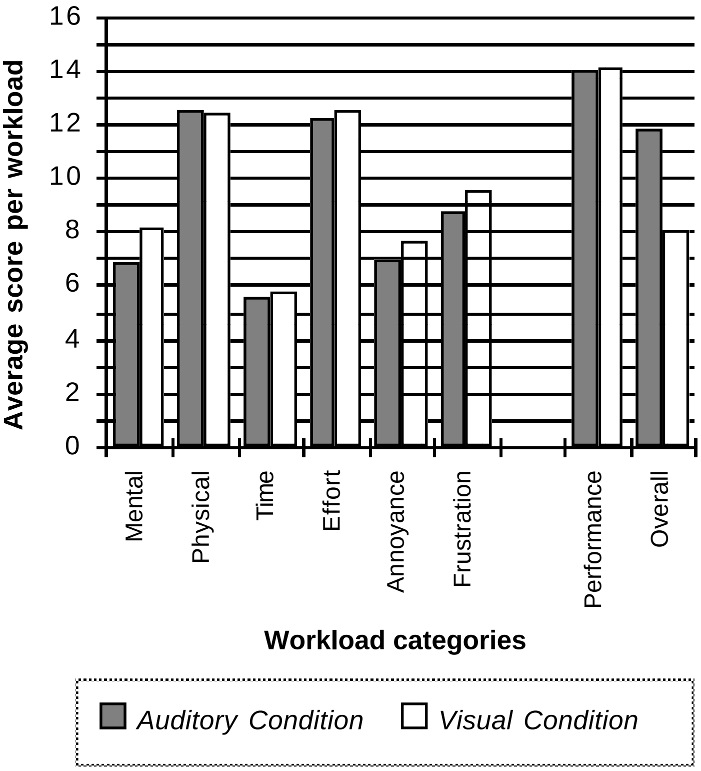


Figure 7.9: *Average TLX workload scores for the auditory and visual conditions of the buttons experiment.*

significantly higher in the overall preference rating (T(11)=5.14, p=0.0003). In this case, the subjects were asked to rate which type of button made the task the easiest. This strongly significant results seems to indicate that the subjects found the task easier with the sonically-enhanced buttons but this did not affect the workload required for the task.

There was no significant difference in terms of annoyance from the feedback in the task. Four subjects rated the auditory condition more annoying than the visual and five rated the visual more annoying than the auditory. This again indicated that the subjects did not find the sound feedback to be annoying. The raw data for the workload analysis are shown in Table 5 of Appendix C.

Error results

After analysis of all the experimental traces from the subjects, four of the slip-off errors recorded by the computer were reclassified as not being errors. These were situations where the subjects had moved off on purpose, using the back-out facility of this type of button.

Figure 7.10 shows the results of error recovery. The raw data are shown in Table 7 of Appendix C. The time to recover from each slip-off was calculated. This was taken as the time from when the slip-off occurred until the user pressed the mouse button down on the correct button again. It was found that subjects in the auditory condition

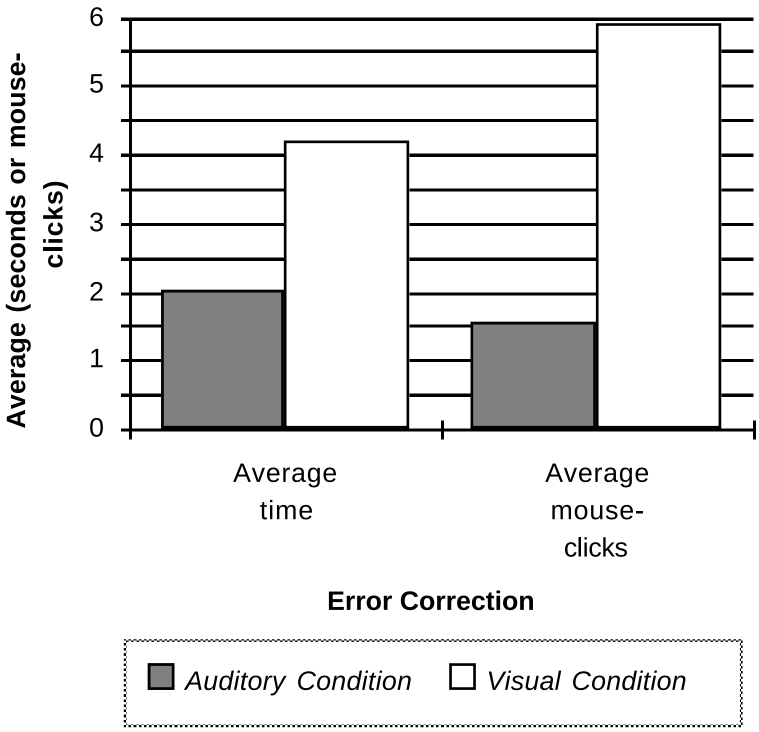


***Figure 7.10**: Error recovery in the buttons experiment. The graph shows average error recovery times and the average number of mouse clicks needed for recovery.*

recovered from slip-off errors significantly faster than in the visual condition (T(12)=3.51, p=0.004). Average recovery times ranged from 2.00 seconds in the auditory condition to 4.21 seconds in the visual condition. The number of mouse button downs and button ups taken to recover from slip off errors was also significantly reduced in the auditory condition (T(12)=4.40, p=0.0008). The average number of clicks to recovery was 1.5 in the auditory condition and 5.89 in the visual. In the auditory condition the subjects recognised an error had occurred and often fixed it by the next mouse button down. In the visual condition it took nearly six button ups and downs before the subjects recovered from an error. These results confirmed the hypothesis that sound can help subjects recover from slip-off errors more quickly.

In this experiment it proved much easier to generate the types of errors required than in the previous one. The auditory condition had an average of 6.6 slip-off errors per subject and the visual condition 3 per subject. There was no significant difference between these scores (T(11)=2.03, p=0.067).

There was no significant difference in the number of background clicks (where the subject missed the button completely) between the conditions (T(11)=1.4, p=0.186). On average, there were 44 background clicks per subject in the auditory condition and 27 in the visual. The auditory feedback did not reduce the number of such errors. The raw data for this analysis are shown in Table 6 of Appendix C.

There was no significant difference in the total number of codes typed in the two conditions (T(11)=0.401, p=0.696). The average number of codes typed per subject in the auditory group was 64.5 and in the visual 65.5. The sounds did not allow the subjects to type codes more rapidly. The raw data for this analysis are shown in Table 6 of Appendix C.

7.6.6 Discussion

The workload analysis showed that there were no significant differences between the conditions on any of the factors. This showed that the sound enhancements did not reduce the workload of the task. However, the subjects very strongly preferred the sonically-enhanced buttons to the standard ones. This may have been because the auditory buttons allowed subjects to recover from errors with less effort. It is unclear why this was not reflected in the workload scores. It may be that recovering from errors was seen as a separate activity from the main task and therefore did not figure in the workload estimates but might have affected preference ratings. It may be that this aspect of workload is not captured in the standard six NASA factors.

The sonically-enhanced buttons did not increase the annoyance or frustration felt by the subjects. This gives further evidence to suggest that if sounds provide useful information they will not be perceived as annoying.

The main hypothesis, that the addition of sound would speed up error recovery, was proved by the experiment. Time to recover from errors and the number of keystrokes needed were both significantly reduced. These results indicate that if sound is used then slip-off problems can be dealt with very effectively. On average, the time to deal with a slip-off error was halved and subjects took just over one mouse-click to recover in the auditory condition. Sound is therefore shown to be a powerful method of dealing with the problem. In the visual condition subjects took longer to recover from errors because they did not look at the area of the screen that displayed the code they had typed, so they did not notice when an error had occurred. They did not see the button feedback because their visual attention moved to the ‘OK’ button when they had pressed a number on the keypad. The interaction required the frequent movement of visual attention from the on-screen keypad to the ‘OK’ button and back; subjects often only looked at the code display area when they had typed in the code and it was then that they found a mistake had occurred. In the auditory condition the subjects heard the button error as it happened and so could correct it straight away. This demonstrated the advantage that can be gained from using a different sensory modality. The task required the subjects to concentrate their visual attention on several different things and this meant that overload occurred. Subjects did not look at the code display area until the whole code had been typed: They needed their visual attention for controlling the mouse. If sound is used then the information presented does not conflict with the visual nature of the task (in the same way as Sellen *et al.'s* footpedal described in the previous chapter) and so does not affect performance of the visual task.

The number of background clicks was not significantly changed by the addition of sound. Many subjects said that they liked the mouse over a button sound because it allowed them to 'multi-task': They could move the mouse from the 'OK' button back towards the keypad and then look at the next code field at the same time. They could then hear when the mouse got to the keypad because when the mouse moved over a button it made a sound.

There was no significant difference between the total number of codes typed in either condition. Even though the subjects in the auditory condition recovered from errors faster they did not type more codes. The subjects in the auditory condition made more slip-off errors than in the visual (although this was not significant). The auditory condition made, on average, 6.6 slip-off errors per subject and each of these took, on average, 2 seconds to recover from, making 13.2 seconds spent on error recovery. In the visual condition there were, on average, 3 slip-off errors per subject taking 4.2 seconds to recover from, making 12.6 seconds spent on error recovery. These two error recovery total times are very similar indicating why there was no difference in the number of codes typed. The fact that the subjects made more errors in the auditory condition wiped-out the advantage gained from recovering from errors more quickly. It was as if subjects became more careless with their clicking because they knew that they could recover from the errors made much more quickly and at little cost. However, recall that the difference in total errors was not significant so no strong conclusions can be made about the number of errors that would occur in a real interface. It is hoped that when sonically-enhanced buttons are used in graphical interfaces users will not make more errors but that they will make the same number of errors and recover more quickly.

In the auditory condition of the scrollbar experiment above, all the graphical feedback was the same as in the visual condition but with sound added. In the experiment describe here some of the visual feedback was removed and replaced with more effective auditory feedback. This feedback was not displayed redundantly with the graphical, it replaced it. This is important because it shows that sound can be used to present information that is currently graphical. The high overall preference for the sonically-enhanced button shows that subjects did not miss the graphical feedback and preferred the auditory.

Justification of the event, status and mode analysis method

The sonically-enhanced buttons were built to overcome the problems identified by the event, status and mode analysis described in Chapter 6. The results show that adding sound in the ways suggested by the method can significantly reduce the time and number of keystrokes needed to recover from slip-off errors. This also lead to a strong preference for the new buttons by subjects. This was due to the fact that they could recover from errors with less cost. These results indicate that the event, status and mode analysis technique identified the problems correctly and suggested a solution that solved them.

In the case of the problem of not being able to identify targets (the mouse over a button sound) there was no advantage from adding sound although many subjects said that they liked the sound for the reasons described above. One reason for the result may have been that it was easy for the subjects to tell what were buttons and what were not in the display because they were all clearly marked and of a large enough size to make them easy to hit.

7.6.7 Future work

The mouse over a button sound could be used for other things. In the current experiment it only indicated that the mouse was over a target. This could be extended to give information about the target. For example, in a hypertext system the sound could give information about the link the mouse was over. It might give information about the size of a file linked to, how far away it was or what class of link it was. This could be encoded into the sound by varying its parameters.

As described in the previous chapter, buttons are used in many areas of the interface, from icons to menus. The advantages demonstrated here could be used in these other widgets to overcome their similar problems. Further experiments could evaluate these.

7.6.8 Summary

The sonically-enhanced buttons were shown to be effective at reducing the time taken to recover from slip-off errors. The number of mouse clicks necessary to recover from slip-off’s was also reduced. The buttons were strongly preferred by the subjects over standard visual ones. These results show that the introduction of such buttons into human-computer interfaces would improve usability. The results again showed that the event, status and mode analysis technique made a correct identification of the problem and the feedback it suggested corrected the problem. This gives further evidence that the analysis technique is a useful one for finding hidden information.

The results also showed that sound could be used to replace visual feedback. In the auditory condition there was no graphical feedback to indicate that the button had been pressed: It was done purely in sound. This proved to be effective and users preferred it. This leads the way to removing other graphical feedback and replacing it with more effective auditory feedback, leaving the visual system to concentrate on the main task the user is trying to accomplish.

7.7 THE EVALUATION OF SONICALLY-ENHANCED WINDOWS

There are many different errors that occur in modern graphical interfaces and one of the most common of these is the ‘unselected-window’ error. This occurs in multi-window systems where the user tries to interact in one window but another is the active one. In Chapter 6 an ESM analysis was carried out to find the problems that caused unselected-window errors. The experiment described here uses the feedback suggestions from this analysis to produce sounds to overcome the problems.

7.7.1 Previous attempts to solve the problem

Unlike the scrollbar and button problems, there have been previous attempts to solve the unselected window problem. The research by Monk [117] and Sellen, Kurtenbach & Buxton [151, 152], described in the previous chapter, worked indirectly towards a solution to this problem with their general work on modes. Reichman’s work, also described in the previous chapter, suggested that colour could be used to solve the problem. There has been one detailed experimental study investigating the problems and this was performed by Lee [105].

Lee used a method first suggested by Harrison & Barnard [90] to prevent unselected window errors. They suggested using a ‘fizzy’ border around the active window. As Lee describes (p 75):

> “The design modifies the window border of the active window from a static state to one that is dynamic by injecting a ‘fizzy’ type of visual activity in the window border after a pre-defined period of inactivity.”

The human perceptual system is very good at detecting changing stimuli and the idea was that when the user looked at the screen his/her attention would be drawn to the rapidly changing border of the active window. In terms of the event, status and mode analysis this feedback was action-independent, demanding, sustained and dynamic. Lee tested this type of feedback and found that it did reduce the number of unselected window errors.

In Lee’s experiment subjects had to answer forty simple questions about buying and selling shares on the stock market. To do this they had to retrieve and display buying and selling share prices. There were two windows: One for the share prices and one to control the experiment. In the control window subjects could choose the ‘buy’ or ‘sell’ prices and then load them into the share price window. For example, to retrieve the buying share prices the subjects had to click on the control window to make it active, click on the ‘buy’ button and then click the ‘share price’ button to load the prices into the other window. Subjects also had to deal with a distractor task. Every 90 seconds another computer behind the subjects interrupted them and they then had to type in a 14 digit random number. When they had done this they could go back to the main task.

The experiment had two conditions. In the first, subjects received static feedback about the active window in the form of a black band surrounding the window. In the second they received the ‘fizzy’ feedback described above. This feedback was activated by 25 seconds of user inactivity.

Lee’s work showed that the number of unselected window errors could be reduced by using dynamic visual feedback. The feedback was especially useful when subjects returned from servicing an interruption. The dynamic feedback caused the user to re-orient to the correct window.

It was decided that a solution to the unselected-window problem would be investigated using sound. As before, using sound would not increase the visual complexity of the display and would not conflict with visual nature of the primary task being undertaken on the computer. An experiment was designed to test sonically-enhanced windows based on the design described above. The aim of the experiment was to see if the event, status and mode analysis suggestions would improve usability. The experiment was a two-condition, within-subjects design, as described in the testing framework above. In one half of the test subjects were given standard visual windows and in the other subjects were given sonically-enhanced ones (see Table 7.1). This design was based closely on that of the experiment by Lee.

It should be mentioned here that due to a fault in the collection of error data in this experiment a detailed analysis of errors could not be conducted. A full analysis of workload was carried out as was an overall analysis of errors. However, this experiment does not have the same status as the two experiments just described. It is included because the design improves upon that of Lee by controlling for certain types of errors (see Section 7.7.4 on unpredictable switches) and it also provides a full workload analysis.

7.7.2 Tasks

The tasks used were similar to those described by Lee. He used a stock exchange simulation but in this experiment a reference database was used. There were two tasks.

The main task was called the *References Task*. The subjects had to answer questions on two references databases. In order to do this they had to switch between two windows. There was also an distractor task, called the *Number Task*, which the subjects had to deal with intermittently whilst carrying out the main task.

References task

Subjects had to answer simple questions from a question sheet about items in one of two bibliographical databases. The databases were displayed in the references window (see Figure 7.11). The figure shows the window containing the human-computer interaction (HCI) database. The other database was on functional programming (FP). Subjects had to search through the databases to find the answers to the questions. To navigate in the references window subjects could use the page up, page down, home and end keys. To swap between databases a subject had to use the control window (see Figure 7.12). To swap from the HCI database to the FP database the subject had to select the control window, press the button marked ‘FP Refs’ and then press the ‘Load Refs’ button. He/she then had to select the references window again to begin searching. The overall layout of the two windows is shown in Figure 7.13. The references were in alphabetical order by author.

The subjects had to answer 40 questions in each condition. Each question indicated which database it was from. The questions were of the form:

1. In HCI Refs: Who was the third author of the article by Berglund & Preis: “Relationship between loudness and annoyance for ten community sounds” ?

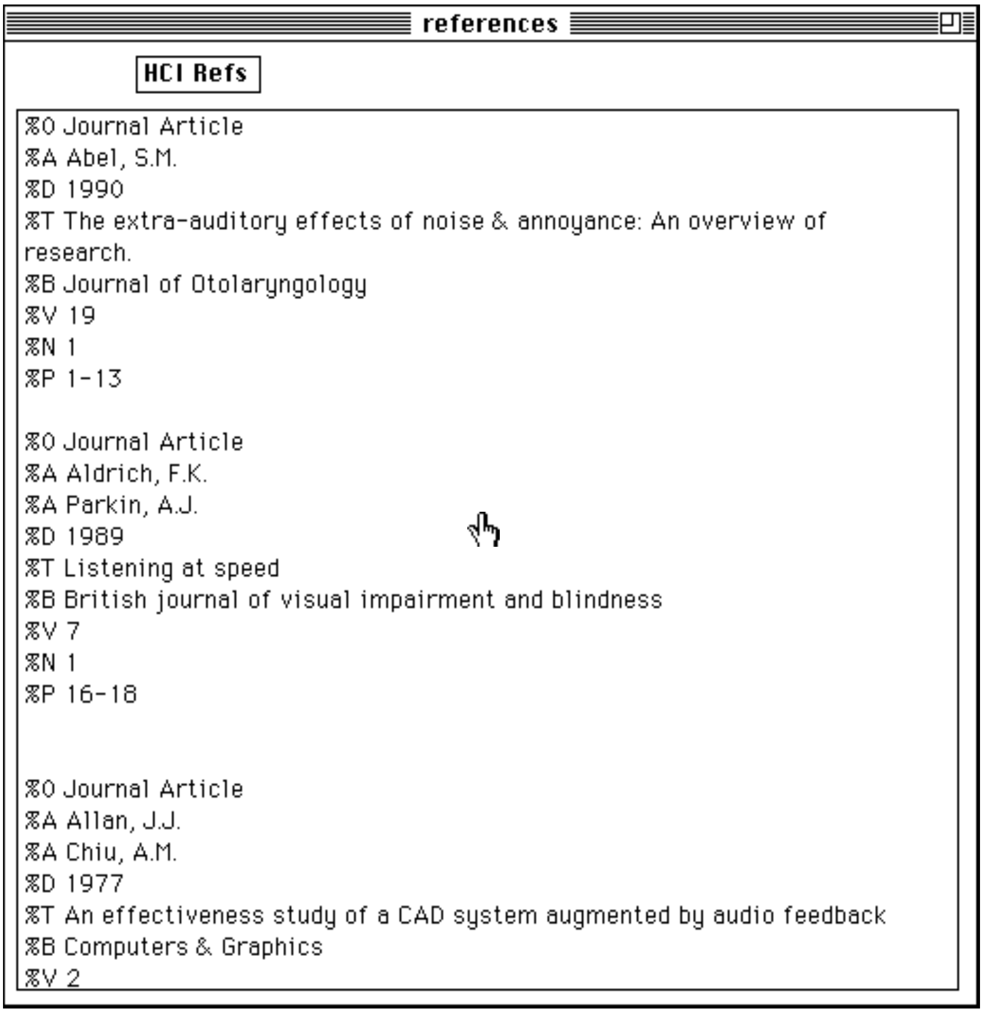
2. In FP Refs: Who published the book by Diller, 1988 ?

3. In HCI Refs: Who published the book by Blattner & Dannenberg in 1992 ?

4. In FP Refs: What are the pages of the book section by Jones, G., 1990 “Deriving the Fast Fourier Algorithm by Calculation” ?

5. In HCI Refs: What are the page numbers of Chapter 9 of the book by Brewer published in 1987 ?

The questions were randomly ordered so that it was not always a question from ‘HCI Refs’ and then a question from ‘FP Refs’. The subjects had to write the answer under the question on the answer sheet. The references were in *Refer* format, a standard bibliographical format (see Figure 7.11). This format was described to each subject at the beginning of the experiment.



%0 Journal Article
%A Abel, S.M.
%D 1990
%T The extra-auditory effects of noise & annoyance: An overview of research.
%B Journal of Otolaryngology
%V 19
%N 1
%P 1-13

%0 Journal Article
%A Aldrich, F.K.
%A Parkin, A.J.
%D 1989
%T Listening at speed
%B British journal of visual impairment and blindness
%V 7
%N 1
%P 16-18

%0 Journal Article
%A Allan, J.J.
%A Chiu, A.M.
%D 1977
%T An effectiveness study of a CAD system augmented by audio feedback
%B Computers & Graphics
%V 2

Figure 7.11: *The references screen of the window experiment showing the HCI references database.*

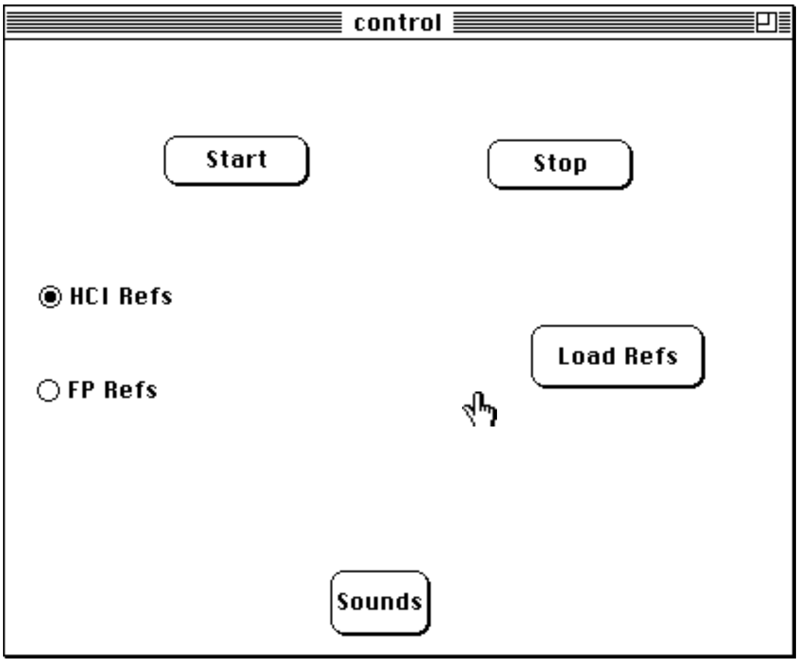


Figure 7.12: *The control screen of the window experiment showing the HCI references database selected.*

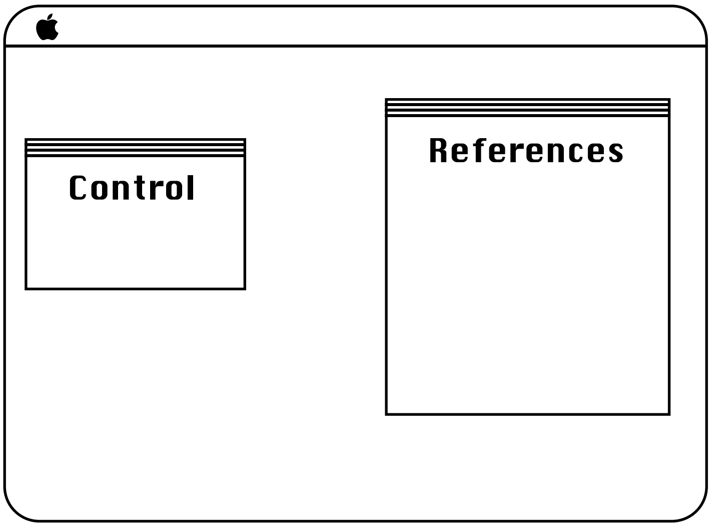
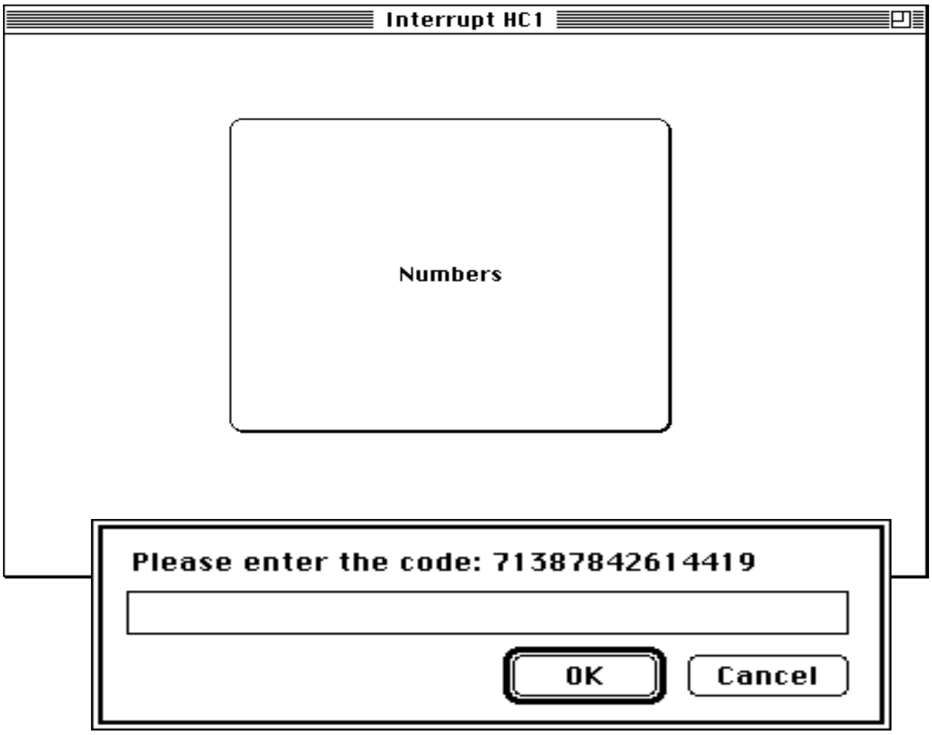


Figure 7.13: *The layout of the control and references windows in the window experiment.*



Please enter the code: 71387842614419

Figure 7.14: *The screen of the number task showing a number to be typed in the window experiment.*

Number task

The number task was used as a distractor and was the same as the one used by Lee in his experiment. Subjects had to press a large button on the screen of a second Macintosh situated behind them when they heard it beep. This happened every 90 seconds. The screen of the number task is shown in Figure 7.14.

They then had to type in a 14 digit random number. When the number button flashed the number had been entered correctly and the subject could go back to the references task. If the system beeped a mistake had been made and the number had to be re-typed. The number had to be typed correctly before the subject could return to the references task. Subjects were instructed to deal with the distractor task as soon as they heard it.

7.7.3 Sounds used

The earcons were designed using the guidelines put forward in Chapters 4 and 5. Figure 7.15 shows the hierarchy of sounds used in the experiment. There were two families of sounds in the sonically-enhanced windows, each based on the same hierarchy. Each window had its own timbre, stereo position and constant background tone. The control window had a violin timbre and a left stereo position, the references window had an electric organ timbre and a right stereo position. These positions mimicked the layout of the windows as shown in Figure 7.13.

The background tone was at a low intensity (just above the ambient threshold) and was at pitch G_2 for the control window and C_4 for the references window. These differences were based on the results from Chapter 5 in presenting sounds in parallel. The different pitches would avoid any chance of confusion. Only one window would be playing a sound at any one time but the techniques in Chapter 5 worked with serial earcons and so would make sure that each of the windows was heard as a separate sound source and not be confused. This feedback was action-independent, avoidable, sustained and static. It gave constant status feedback about which was the current active window. This had to be avoidable so that subjects could choose not to listen to it so that it did not become annoying.

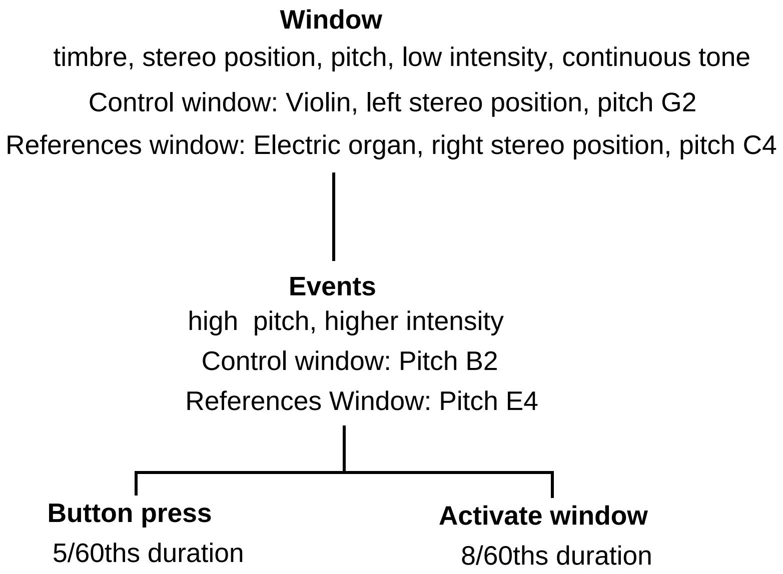

Window
timbre, stereo position, pitch, low intensity, continuous tone
Control window: Violin, left stereo position, pitch G2
References window: Electric organ, right stereo position, pitch C4
Events
high pitch, higher intensity
Control window: Pitch B2
References Window: Pitch E4
Button press
5/60ths duration
Activate window
8/60ths duration

Figure 7.15: *The hierarchy of earcons used in the window experiment.*

In Section 6.6.7 in the previous chapter it was suggested that action-dependent feedback should be used to complement the action-independent and reinforce the status feedback from the active window. When a key was pressed in the references window or a button clicked in the control window action-dependent, static, transient and demanding event feedback was given. It was hoped that the combination of both transient and sustained sounds would help reinforce knowledge of the active window. In the control window when any of the screen buttons were pressed a short, higher intensity sound was played. In this case it was a violin note at B_2 for 5/60 sec. When any one of the navigation keys were pressed in the references window an organ note at pitch E_4 was played for 5/60 sec. This was similar to the feedback Monk [117] used to indicate different modes in his keying-contingent sound experiment described in the previous chapter.

Audio feedback to overcome the problem of discontinuity of input was provided. The event of switching from one window to another was marked by a short, high-pitch note in the new window's timbre at increased intensity. For example, when swapping to the control window a violin note of pitch B_2 was played for 8/60 sec. When swapping to the references window an organ note of pitch E_4 was played for 8/60 sec. This demanding sound gave feedback that an event had taken place. The sound then returned to the low intensity constant status tone. This event sound was action-dependent, demanding, static and transient.

It was hoped that providing both action-dependent and independent feedback would give an 'intermediate' level between avoidable and demanding. If only a constant background sound was used it would have to be demanding to indicate the current window effectively. This, however, would have quickly become annoying for users. If only action-dependent sounds were used then when subjects came back to the system they would get no extra help in re-orienting to it until they hit a key, which may have meant an error had already been made. The combination of both of these methods provided a non-intrusive constant background sound plus demanding keystroke sounds to reinforce the current window.

7.7.4 Experimental design and procedure

The experiment was in two halves and was based on the design from the experimental testing framework described above (see Table 7.1). The layout of the experiment was described in the section on tasks above. To start the experiment subjects pressed the 'Start' button shown in Figure 7.12. When they had completed the 40 questions they pressed the 'Stop' button. The 'Sounds' button was used to turn on the sounds in the auditory condition.

The distractor, or number, task interrupted the subject every 90 seconds. The interruption was signalled by constantly repeating groups of two beeps on the second machine. These beeps continued until the subject clicked the number button. Subjects were instructed to deal with this task as soon as they heard the beeps and stop what they were doing on the references task.

Screen saver

A screen saver cut in when there was no activity in either of the windows for 25 seconds. This blacked out the screen so that neither of the windows could be seen and turned off all of the sounds. The sounds had to be turned off otherwise the subjects would still be able to hear the active window when dealing with the distractor (due to the omni-directional nature of sound). This was timed such that when a subject turned to the second computer to deal with the interruption task he/she returned to the main machine to find the screen saver on. The subject then had to click the mouse to turn off the screen saver. In the auditory condition the sounds came on again as the screen was re-drawn. The idea behind the screen saver was that it, combined with the distractor task, simulated the ‘cup of tea problem’: When the user turned back to the system after dealing with the distractor task the screen was blank. The subjects had to re-orient to the system. Turning on the graphics and the sound together allowed the synchronisation of the start of both in the auditory condition. The sound for the currently active window started up with a demanding note (the same sound as switching from another window). It was hoped that this would allow the user to re-orient to the system more accurately.

Unpredictable switches

In order to test the subjects’ ability to re-orient themselves to the system after the distractor task it was necessary to make sure that they were not just remembering which window was active and carrying on from where they left off (Lee did not control for this in his experiment). Therefore three *unpredictable switches* were put into each condition. In this case, when such a switch occurred the system returned from the screen saver into a different window from the one it was left in. So, for example, if the subject dealt with the distractor task when he/she was in the control window the system would return in the references window. If the subjects were just remembering where they were and not using the feedback they would make unselected-window errors. These switches happened at fixed points in the system. One occurred near the beginning of the task, one in the middle and one near the end.

Errors

The errors were collected in a similar way to that of Lee. A trace of data for each subject was stored by the system. It was hoped that the references task would cause errors of discontinuity of input. The users had to swap between the two windows in a random way changing their focus of attention. They would forget to click the mouse in the window they wanted. The distractor task would cause the ‘cup of tea’ variant of the error where, after a pause, the user re-oriented to the wrong window.

An unselected window error in the control window was signalled in the trace by:

- ❖ a click on the location of one of the radio buttons (HCI Refs or FP Refs)
- ❖ a click on the Load Refs button
- ❖ a click on the actual radio button
- ❖ a click on the Load Refs button

In this case the subject tried to select a references database and load it before selecting the window. The first click would select the window rather than a new database. The click in the ‘Load Refs’ button would just load the same database as was there already. An unselected window error in the references window would be shown by pressing any of the scrolling keys whilst in the control window. These data were recorded by the system automatically into a log file.

Training

Before the first condition in the experiment subjects were trained on the task. They were given eight training questions of the type they would get in the real experiment. They were allowed to do the first four of these without the distractor task on, the final four were done with the distractor so that they could get used to both tasks. Before the second condition more training was given so that the subjects could get used to the new interface. Again half of the training was done without the distractor task.

Visual Condition

Subjects saw standard Macintosh windows with no sound added. To start the experiment the subject pressed the ‘Start’ button in the control window and when they had answered all 40 questions they hit the ‘Stop’ button. The questions in each of the condition (and each of the training sessions) were different.

Auditory Condition

Subject saw the standard Macintosh windows again but this time the auditory feedback described above was given as well. In the initial training of subjects for this condition the feedback provided by the window was described in detail. The training and testing then proceeded as described above for the visual condition.

7.7.5 Experimental hypotheses

The hypotheses were based around the predictions of the ESM analysis technique described above. There should be fewer discontinuity of input errors in the auditory condition because the subject received more feedback about changing windows and also more feedback from the current window. There should also be fewer ‘cup of tea’ errors because on resumption after a distraction there was more salient feedback about which was the active window. Subjects in the auditory condition should not make mistakes with the unpredictable switches because the salient feedback would make them re-orient to the correct window. Time to complete the tasks should be less in the auditory condition because fewer errors should mean that less time was spent in error recovery.

The workload felt by subjects should be reduced because the extra feedback would make it easier to recover from errors. Physical demand and time pressure will be unaffected as they were unchanged across conditions. There will be no increased frustration or annoyance due to the addition of sound as the auditory feedback will provide information that the subjects need.

7.7.6 Results

TLX results

Figure 7.16 shows the average scores for each of the workload categories plus the two extra ones. The raw data for these analyses are shown in Appendix C Table 8. Paired T-tests were carried out on the auditory versus visual conditions for each of the workload categories. Overall raw workload (based on the standard six factors) was 10.5 for the auditory condition and 10.8 for the visual. These scores were not significantly different (T(5)=1.74, p=0.141).

There were no significant differences between any of the factors. There was no difference in overall preference as there had been in the two previous experiments (T(11)=1.148, p=0.274). Annoyance in the auditory condition was not significantly different to that in the visual condition (T(11)=1.288, p=0.224).

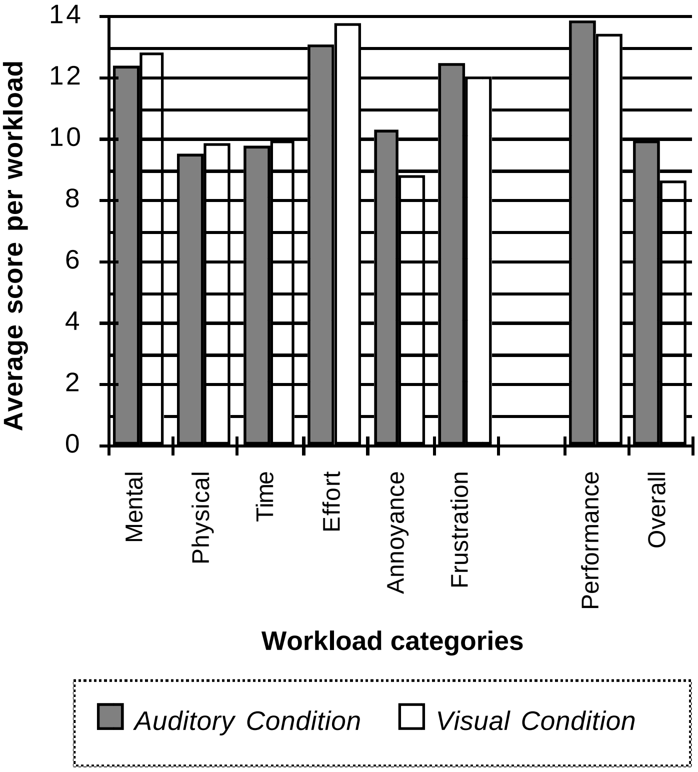


Figure 7.16*: Average TLX workload scores for the auditory and visual conditions in the window experiment.*

Timing and error results

Figure 7.17 shows the unselected window errors for each window. The raw data are shown in Appendix C Table 9. In the references window the number of errors was reduced from 36 in the visual condition to 22 in the auditory although this differences was not significant (T(11)=1.37, p=0.223). These figures are for the number of different occurrences of errors. Often more than one error was made at each occurrence. The overall number of errors in the references window was 57 for the auditory condition and 92 for the visual. The scores were not significantly different (T(11)=1.37, p=0.195). This is the number of errors made before the subjects realised that the wrong window was selected. This information was only recorded for the references window. In the control window errors followed the pattern described in the section on errors above.

The overall time to complete the tasks was also recorded. On average a subject in the auditory condition took 1574.7 seconds and 1510.5 seconds in the visual. These times were not significantly different (T(11)=1.33, p=0.208). The raw data are shown in Appendix C Table 9.

Due to an error in the program recording the data for the experiment it was not possible to get a breakdown of the errors which meant that the analysis could only be done at a high level. Unfortunately it was not possible to find out how many of the unselected window errors were due to the subjects returning from the distractor task and how many were due to discontinuity of input.

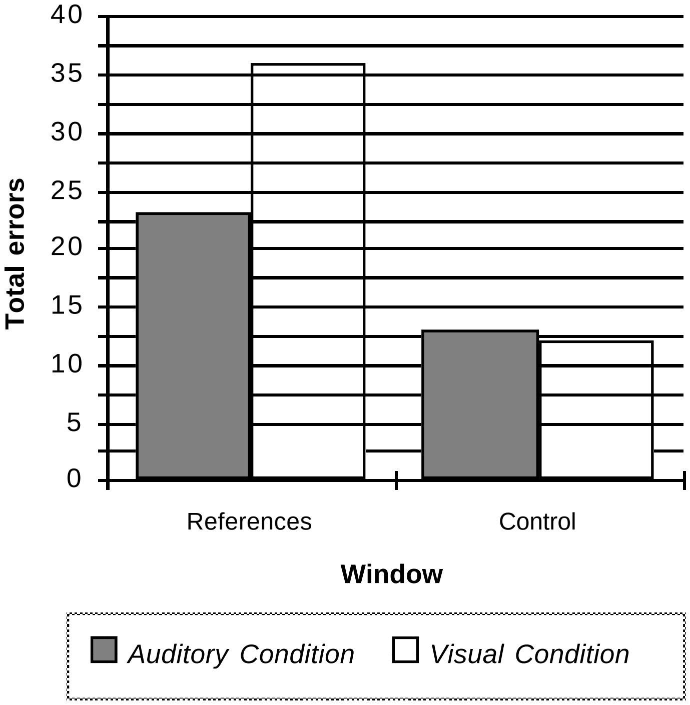


Figure 7.17: *Total unselected window errors in the window experiment.*

7.7.7 Discussion

The workload results showed no differences between the two conditions. None of the individual workload scores were significantly different. This indicated that the sonically-enhanced interface did not lower the workload of the task. Unlike the previous two experiments there was also no significant difference in overall preference; subjects did not prefer one interface to other the other.

Subjects did not find the sonically-enhanced windows more annoying than the standard visual ones. This confirms the results of the previous two experiments. Even though potential users of auditory interfaces claim that they would not like them because they would be annoying the results do not show this to be the case.

The results showed that unselected-window errors were being made by subjects in both conditions. In the references window there were a third less errors in the auditory condition than in the visual but this difference was not significant. Due to the problems of data collection it is impossible to look at this decrease in more detail to see if the overall scores were hiding any differences.

There were more errors in the references window than the control window. This was probably due to the subjects spending more time in the references window: They had to spend time searching for answers to the questions. In the control window all they had to do was press two buttons to load the other references database. They were therefore interrupted by the distractor task more in the references window. More discontinuity of input errors were also likely to occur in this window. These occurred because, for example, a subject activated the control window to load the other database, but then failed to re-activate the references window again when switching attention back to it. When subjects switched to the control window they often remembered to click in it because it was a special occurrence. When they returned to the references window to search for the next reference they often forgot to click in the window. This may have been due to them thinking that it was the active window because most often it was the active one. Another reason for more errors in the reference window may have been that subjects only used the scrolling keys when in it and not the mouse. Therefore, when wanting to use the references window, they started using the scrolling keys straightaway rather than using the mouse to select the window. The problem with data collection meant that the types of different errors could not be identified.

From the analysis available it was difficult to say if the sonically-enhanced windows were having an effect. There were fewer unselected window errors in the auditory condition but neither the workload scores or time taken were different between the conditions.

Justification of the event, status and mode analysis method

The sonically-enhanced windows were designed to overcome the problems identified by the event, status and mode analysis. With only the limited amount of data analysis that could be performed it is difficult to tell if the results justify the analysis. The workload for the task in the auditory condition was not lowered but the number of unselected window errors showed some signs of decrease. Further investigation would be needed to see if the solutions suggested by the analysis actually solve the problems.

7.7.8 Future work

The experiment needs to be run again with the error in data collection corrected. The results showed that unselected window errors were being generated (up to 92 in the visual condition) so that the design was effective.

7.7.9 Summary

Sonically-enhanced windows were tested and shown not to reduce the workload in the experiment. There was no overall preference for the new windows. The was, however, no increased annoyance due to sound. Constant background sounds were used for feedback in the windows and these had the potential to be annoying but the results have shown this not to be the case. This means that such continuous sounds can be used to present status information in interface without problems of annoyance. Due to problems in the experiment it is unclear as to whether the auditory feedback reduced the number of unselected window errors. Subjects did not perform any faster but there was some indication that there may be fewer errors. The auditory feedback did not make the subjects perform any worse.

7.8 GENERAL DISCUSSION

7.8.1 New earcon guidelines

From the experiments described here and other research some further guidelines for the use of earcons have been identified. The earcons used in this chapter were much simpler than those in Chapters 4 and 5. The previous chapters showed that complex earcons could be recognised by listeners. The earcons here show that simple, practical earcons can be designed based on the guidelines produced earlier. Because the earcons here were simpler some different guidelines emerged when they were being created.

When designing a family of earcons start with timbre, rhythm and register. These can be used to create the basic structure. For example, each family of earcons might be given a different timbre. The family might also have a default register and spatial location. Rhythm can then be used to create the major sub-groups within each family. These can also be differentiated by pitch, intensity, chords or effects such as chorus or delay. Care should be taken to make sure that the earcons are recognisably different. Some general information about the main parameters follows. These guidelines include those from Chapters 4 and 5 and some new ones generated after creating more earcons.

- ❖ *Timbre*: Use musical instrument timbres (see Section 4.4.1). Where possible use timbres with multiple harmonics as this helps perception and can avoid masking (see Sections 2.3.1 and 5.2.2). Timbres should be used that are subjectively easy to tell apart. For example, on a musical instrument synthesiser use ‘brass’ and ‘organ’ rather than ‘brass1’ and ‘brass2’. However, instruments that sound different in real life may not when played on a synthesiser, so care should be taken when choosing timbres. Using multiple timbres per earcon may confer advantages when using compound earcons (see Section 4.6.3). Using the same timbres for similar things and different timbres for other things helps with differentiation of sounds when playing in parallel (see Section 5.2.2).

There is also another reason why care must be taken when timbres are chosen. Some timbres are continuous and some are discrete. The electric organ and violin timbres are continuous: They carry on until they are turned off. Piano or drum sounds are discrete: They only last a short time. This is the nature of different musical instruments. If continuous sounds are needed to sonify an interaction then discrete sounds would have to be constantly turned on and off if they were to be used. This can limit the choice of available timbres.

- *Pitch*: Do not use pitch on its own unless there are large differences between those used (see Section 4.4.1). Complex intra-earcon pitch structures are effective in differentiating earcons if used along with rhythm (see Section 4.5.1). Some suggested ranges for pitch are (from Patterson [128]): Maximum: 5kHz (four octaves above C_3) and Minimum: 125Hz - 150Hz (the octave of C_4).

- *Register*: If listeners must make absolute rather than relative judgements of earcons then pitch/register should not be used (see Section 4.4.1). A combination of pitch and another parameter would give better performance (see Section 4.6). If register alone must be used then there should be large differences between earcons but even then it might not be the most effective method. Two or three octaves difference give better rates of recognition (see Section 4.6.1). This is not a problem if relative judgements are to be made.

- *Rhythm and duration:* Make rhythms as different as possible. Putting different numbers of notes in each rhythm is very effective (see Section 4.6.1). Patterson [128] says that sounds are likely to be confused if the rhythms are similar even if there are large spectral differences. Small note lengths might not be noticed so do not use notes less than sixteenth notes or semi-quavers (see Section 4.6.1). This depends on the tempo. If 180 bpm is used then sixteenth notes last 0.0825 sec.

  If the sounds used are simple, for example just indicating events, then durations can be less. In the sonically-enhanced widgets experiments (this chapter) the duration of the sounds was reduced to only 0.03 seconds. These short durations were shown to be usable and easily recognisable by listeners because they only communicated one thing (for example, a successful screen button click). These short sounds help the earcons keep up with pace of interactions (as does playing sounds in parallel).

- *Intensity*: Although intensity was not examined here some suggested ranges (from Patterson) are: Maximum: 20dB above threshold and Minimum: 10dB above threshold. Care must be taken in the use of intensity. The overall sound level will be under the control of the user of the system. Earcons should all be kept within a close range so that if the user changes the volume of the system no sound will be lost (see Chapter 2 and Section 3.5). Intensity does not have to be used to make earcons attention grabbing.

- *Spatial location:* This may be stereo position or full three-dimensions if extra hardware is available. This is very useful for differentiating multiple sounds playing simultaneously (see Sections 2.6.2, 5.2.2 and 5.3). It can also be used with serial earcons, for example each application might have a different location.

- *Compound earcons*: When playing earcons one after another use a gap between them so that users can tell where one finishes and the other starts. A delay of 0.1 seconds is adequate (see Section 4.5.1). If the above guidelines are followed for each of the earcons that is to be combined then recognition rates will be high.

- *Making earcons demanding:* The event, status and mode analysis technique requires sounds to be demanding. This can be achieved in different ways. It can be done by using intensity. This is effective but potentially annoying for the primary user and others nearby [18, 166]. Rhythm or pitch can be used (perhaps combined with lower intensity) instead because the human auditory system is very good at detecting changing stimuli (see the sounds used in all three experiments in this chapter). If a new sound is played, even at a low intensity, it will grab a listener's attention. Alternatively, if the rhythm of a sound is changed (perhaps speeding up or slowing down) this will also demand attention.

  Further techniques for making sounds demanding are to use: High pitch, wide pitch range, rapid onset and offset times, irregular harmonics and atonal or arrhythmic sounds (for more see [64, 65]). The opposites of most of these can be used to make sounds avoidable but in this case the main things are low intensity and regular rhythm.

This new set of guidelines improves on the ones from Chapters 4 and 5. It can now be used as part of the structured method for adding sound to user interfaces.

7.8.2 Earcons in a real system

The sounds in the window experiment simulated how auditory feedback might be used when sonifying an overall system. Each window had auditory feedback based on the same hierarchy but used different timbres and spatial locations. The event sounds within each window used the same structure but were based on the timbre and spatial location of the window. In a real example, each application would have a different timbre and spatial location, all the widgets within it would use this information and then add rhythm or pitch as necessary. This is shown in Figure 7.18. At Level 1, the three applications all have different timbres and spatial locations. These are inherited by Level 2. The widgets then modify these with pitch, rhythm, etc. and these are constant across applications. This approach has now been tested for the first time. This hierarchy is very similar to that used for the earcons in Chapters 4 and 5.

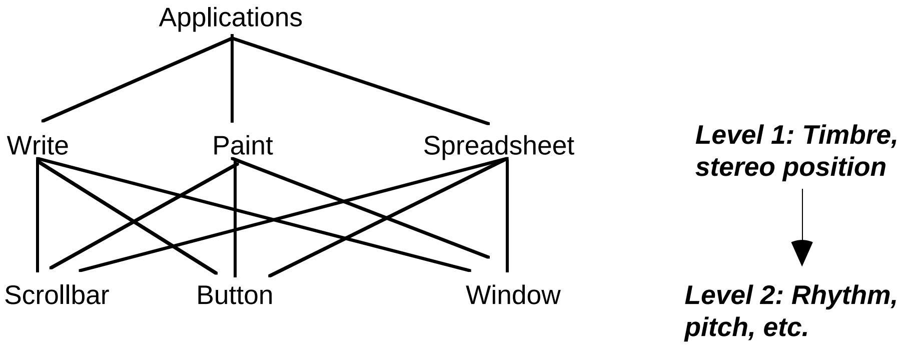

Applications
Write
Paint
Spreadsheet
Scrollbar
Button
Window
Level 1: Timbre, stereo position
Level 2: Rhythm, pitch, etc.

Figure 7.18*: A hierarchy of earcons across applications.*

This means that the simpler earcons used in the experiments in this chapter build up into the more complex hierarchies described in Chapters 4 and 5. Care must be taken to make sure that all of the sounds fit together. In the case of the scrollbar, its continuous tone must be made to fit with the continuous tone from the application as a whole. This could be done by making the continuous tone from the application the position in document sound. It would then change as the user scrolled through the document. Alternatively, the continuous sound from the window could be made to fade into the background by reducing its intensity and the continuous scrollbar sound could take its place. This would act in the same way as the window sound, giving status information about the window by its timbre and spatial location. The scrolling up and down sounds would be mixed in with the continuous sound as they were in the experiment described above.

The screen button earcons would work in the same way as the scrollbar ones. The mouse over a button sound would be played at a different pitch to the continuous window sound and slightly louder so that it stood out but did not become annoying. The mouse button down sound would be louder and so stand-out above the background. The success sound would be mixed in with the continuous sounds, as in the scrollbar. A slip-off could, as before, result in silence with all the sounds being turned off for a short time. Other sonically-enhanced widgets could be used in the same way.

7.8.3 The structured method for integrating non-speech audio into human-computer interfaces

This chapter has shown that if the earcon guidelines are used with the event, status and mode analysis technique then effective sounds can be added to an interface to improve usability. This structured technique could be used by a designer who has no skill in sound design to add useful sounds to an interface. This is a major step forward in the use of sound at the interface. Many systems that have used sound so far have just demonstrated that it is possible. This work goes further; it gives a designer a way of adding effective sounds. He/she can use the event, status and mode technique to find where hidden information might exist in an interface. The predictions the technique makes for the feedback needed can be used with the earcon guidelines to produce the necessary sounds to make it explicit. It is currently the only method for doing this.

7.9 CONCLUSIONS

The experiments described above have shown that auditory enhanced widgets can improve usability by increasing performance and reducing the time taken to recover from errors. The results have shown that the predictions of hidden information and the suggested solutions made by the event, status and mode analysis technique were effective. The technique is thus a powerful tool for a designer sonifying an interface. An interface designer could use the structured method based on the earcons guidelines from Chapters 4 and 5 along with the event, status and mode analysis technique from Chapter 6 to investigate an interaction, find the problems due to hidden information and add sound to overcome them. This is a step forward from previous *ad hoc* methods where the designer had to guess where to add the sound and what type of sound would be best. The sounds added by following this structured method would be consistent, effective and improve usability.

The testing framework described was shown to be effective at evaluating sonically-enhanced widgets. This could be used in the future to investigate further widgets.

Some of the problems of annoyance have been dealt with. Annoyance due to sound for the primary computer user has been shown not to occur in any of the three experiments described here. Both continuous and discrete sounds of different pitches and intensities have been used and in none of the experiments was there a significant difference in the amount of annoyance felt by subjects in the auditory condition. This was because the sounds used provided information that the user needed, they were not gimmicks. In two of the experiments, rather than perceiving the sounds as annoying, subjects significantly preferred the auditory enhanced-widgets over the standard visual ones. However, this is only one half of the problem; an investigation is still needed to find out the annoyance that might occur to colleagues working nearby.

In the button experiment sound was productively used to replace some of the visual feedback present in the standard button. This leads the way to removing other graphical feedback and replacing it with more effective auditory feedback, reducing the clutter of the visual display and leaving the visual system to concentrate on the main task the user is trying to accomplish.

The results of the work described here showed that widgets which combined both auditory and visual feedback were more effective than purely visual ones because they made use of the natural way that humans deal with information in everyday life, combining multiple senses to give complementary information about interactions. It is now clear that sonically-enhanced widgets should be added to standard graphical interfaces to improve usability. The structured method for adding sound has also been shown to be effective.

CHAPTER 8: CONCLUSIONS

8.1 INTRODUCTION

The final chapter summarises the work described in this thesis and the results achieved. It discusses some of the limitations of the work and how they could be overcome. It suggests areas for future investigation of sound in the interface and concludes by assessing the contribution of the thesis to the area of auditory interfaces.

8.2 SUMMARY OF RESEARCH CARRIED OUT

A summary of the work carried out and results achieved is given below. The research is described in terms of the two questions that the thesis addresses and the overall application of the structured method.

8.2.1 What sounds should be used at the interface?

In Chapter 4 a detailed investigation of earcons was described to answer the question of what sounds should be used at the interface. The experiments carried out investigated whether earcons were better than unstructured sounds and whether musical timbres were better than simple tones. An experiment was conducted with earcons designed using the rules described by Blattner *et al.* [25] and musical earcons using synthesised musical instrument timbres. There were four phases to the experiment. The results showed that there were no overall differences between the groups. This indicated that earcons were not communicating their information effectively. The highest overall recognition rate was only 58%. A more detailed analysis showed that in the first phase there were problems with the recognition of the rhythms used. However, the musical timbres were shown as being significantly easier to recognise than the simple tones suggested by Blattner *et al.* (over 70% recognised compared to less than 40%). In the second phase better recognition of rhythm led to overall scores of up to 75%. In this phase the earcons were significantly better recognised than the unstructured sounds. The final phase tested the combination of two earcons. The scores were similar to phases one and three.

A second experiment was carried out to correct the problems highlighted in the first. More distinct rhythms were chosen and a short gap was put between the two parts of combined earcons. The overall recognition rates of 75% were significantly better than the previous experiment. The recognition of rhythm was greatly improved, reaching the level of timbre. The recognition of combined earcons rose to 65%. The two experiments also investigated the effect of musical skill on earcon recognition. It was shown that if musical earcons were used then non-musicians performed as well as musicians.

In Chapter 5 a method for increasing rates of presentation by playing earcons in parallel was investigated. Results showed the parallel compound earcons were not significantly differently recognised to the standard serial compound earcons. On the first presentation of the earcons subjects scored around 80% in both conditions and on the second presentation scores rose to 85%-90%. These results suggested that parallel earcons were a good method for rapidly communicating complex information in sound.

From the knowledge gained in these experiments a set of practical guidelines was created for earcon designers. Those proposed by Blattner *et al.* were shown to be too subtle to be effective. These new guidelines also combined knowledge of existing auditory warning design and psychoacoustics (described in Chapter 2).

8.2.2 Where should sounds be used at the interface?

Chapter 6 proposed an analysis technique for finding where to use sound at the human-computer interface. This technique provided an informal way of modelling interactions to reveal hidden information. Hidden information was chosen to be presented in sound because it can cause errors. Interactions were modelled in terms of event, status and mode information. The technique was used to investigate some basic interface widgets. It was applied by analysing a generic (ideal) interaction to identify all the event, status and mode information. This was then categorised in terms of the feedback required to present it. A real interaction was analysed in the same way. If there was any information not in the real one then this was hidden information. The technique was used to investigate several interactions and suggest sounds to overcome the problems.

The first investigation was of a caps-lock key. The problem here was that users might not notice that they were in upper-case mode because the event of changing from one mode to another was avoidable and status feedback from the caps-lock key was itself avoidable. To overcome this, the analysis technique suggested adding a demanding event sound for when the caps-lock key was pressed. An action-dependent, demanding sound would also be played for each character typed when in upper-case mode to indicate the status of the key.

Next, the problems of users slipping off screen buttons was investigated. This occurs because the feedback from a correct button press is identical to an incorrect one. The analysis suggested that a demanding sound should be added to the event of moving off a button so that the user knew he/she had moved off. A demanding sound should also be added to indicate a successful press of the button.

Problems of kangarooing and loss of position were investigated in a scrollbar. If a demanding, action-dependent event sound was added when scrolling window by window then users would be able to tell if kangarooing had occurred. If a demanding event sound to mark the crossing of a page boundary was combined with an avoidable, sustained status sound for each page then loss of sense of position in a document can be avoided.

The final widget analysed was the window. Often users interact with the wrong window by mistake leading to the ‘unselected-window’ error. To overcome this a demanding event sound should be added for when the user changed from one window to another. To improve the recognition of which was the active window when the user returned to the computer, avoidable, continuous status feedback should be given.

8.2.3 Application of the structured method

Chapter 7 of the thesis described the structured method for integrating sound into human-computer interfaces. This method was a combination of the analysis technique from Chapter 6 and the earcon guidelines from Chapters 4 and 5. The chapter investigated if the predictions of where to use sound from the analysis technique could be combined with the earcon guidelines to produce effective sonically-enhanced widgets that improved usability. Three widgets were investigated: Scrollbars, buttons and windows. A testing framework was created so that they could all be tested in a consistent manner. To investigate if usability was improved time, errors and workload were measured.

Sound was first integrated into a scrollbar. The hypotheses of this experiment proposed that in the auditory condition time to recover from errors would be reduced as would workload and the total time to complete the tasks. The workload results showed that mental demand was significantly reduced in the sonically-enhanced scrollbar. Subjects also preferred the sonic scrollbar to the visual one. In one task in the auditory condition, there was a significant reduction both in the overall time taken and the time taken to recover from errors.

The next widget investigated was the sonically-enhanced button. The hypotheses again proposed that subjects would recover from errors more quickly and workload would be reduced. The results in this case showed no differences in terms of workload between the two conditions. There was, however, a strong preference for the sonically-enhanced buttons. Subjects in the auditory condition also recovered from errors significantly more quickly than those in the visual.

The final experiment investigated sonically-enhanced windows to try and overcome unselected-window errors. The hypotheses proposed that there should be fewer unselected window errors in the auditory condition and both time to complete tasks and workload would be reduced. The results showed no differences in workload. There were no overall differences in the number of errors between conditions and the overall time to complete the tasks were not significantly different. Due to an error in the software recording the data for the experiment it was impossible to perform a detailed analysis of the error data.

In all three experiments, annoyance to the primary computer user was specifically investigated. In none of the sonically-enhanced widgets was the annoyance rated significantly differently to the standard visual ones.

8.3 LIMITATIONS OF THIS RESEARCH

The approach taken to integrating non-speech sounds into user interfaces has been shown to be effective. However, it has some limitations that should be considered. These limitations are described in terms of the two research questions and the application of the structured method.

8.3.1 What sounds should be used at the interface?

Earcons were investigated in detail as they had never before been evaluated. One of the main advantages claimed for earcons by Blattner *et al.* was that they could be built up into complex messages by simple combination. For earcons to be effective at the interface high rates of recognition were needed. The two experiments in Chapter 4 showed only low recognition (only 65% for compound earcons, even after the improvements made in Experiment 2). This would limit the usefulness of earcons. However, the experiments were ‘worst-case’ analyses in order to rigorously test earcons. By simply improving the training rates of up to 90% were achieved in the parallel earcon experiment in Chapter 5. This meant that earcons were an effective method of communicating information in sound.

When sonifying an interface it is likely that earcons would be used in combination with auditory icons. These are intuitive in some situations so advantage should be taken of this. The thesis did not investigate the best ways to combine the two systems. A decision was made to use earcons and nothing else. Fitch & Kramer [68] performed the first evaluation of such a system combining the two sound methods and showed it to be effective. Further work would investigate the ways to combine the two sound systems.

The work in this thesis did not test recognition of earcons over time. In Experiments 1 and 2, phase III was a re-test of the phase I earcons. It showed that they could be remembered over a short time (approximately 15 minutes). In a real interface earcons might have to remembered over much larger intervals (perhaps weeks). It is not clear if they could be remembered in this way, if they cannot then their usefulness would be significantly reduced. Further work is needed to investigate this.

In the parallel earcons experiment described in Chapter 5, only recognition rates of earcons were measured. It may be that although recognition rates were high, the workload required to use the earcons would be increased. The earcons were much more complex, listeners having to attend to two sound sources at once. Such earcons may require more cognitive resources to recognise meaning that subjects perform less well on other tasks. Alternatively, if subjects had to use parallel earcons whilst performing other tasks then recognition rates might fall. Workload was not tested as part of the parallel earcons experiment to find out if they were harder. It is hoped that, as the human auditory system constantly processes multiple sounds simultaneously, parallel earcons would not be a problem. Further investigations would be necessary to see if this was the case.

One further reason for not using parallel earcons is that ordinary serial earcons could just be played faster. The work in Chapter 5 did not investigate this to see where rates of recognition broke down when speed was increased. It could be that the same recognition rates as parallel earcons could be maintained by playing the serial earcons faster. However, it is likely that doubling the speed of serial earcons would lower recognition. This could be corrected by more training but parallel earcons succeeded with the same amount of training as the serial form. The experiment also showed that earcons could be played simultaneously and this ties in with plans for future research.

The earcon guidelines produced were very general and still require knowledge of music and psychoacoustics to use them effectively. The guidelines were supposed to contain all the knowledge an interface designer not skilled in sound design would need to create earcons. At the current time this is not possible because only a few experiments have investigated earcons. More work is needed to provide a complete set of guidelines.

8.3.2 Where should sounds be used at the interface?

Chapter 6 dealt with the question of where to use sound at the interface. It did not investigate all of the different types of information that sound could represent. Alty & McCartney [4] have shown that there is a need to find the best ways of combining sound and graphics. This thesis chose to display hidden information in sound. It showed that sound could be used in this way but it may not be the best use of sound. For example, an alternative could have been to add sound to all events and make all status information visual. This idea would fit well with the previous uses of sound (as alarms and warnings). There is much research still to do in this area but the thesis has shown that it is possible to add sound to display this one type of information so it should be possible to display others.

The analysis technique made a blanket assumption that all hidden information should be made explicit. This may not always be the case. There may be some information that the user never needs to be aware of. Displaying it would unnecessarily clutter the multimodal interface. The analysis technique is simple and this makes it easy to use but this may mean that it suggests adding sound to make something explicit that the user does not need to know. Further analyses would have to be carried out to discover if this was a problem.

In some cases the current visual feedback could be redesigned to reveal any hidden information. The might mean that the problems could be solved without making the visual display more complex, which was the motivation for using sound in the first place. There would then be no need for sound. However, this would not work in all cases (for example, with the screen button) because the auditory system has different properties to the visual and these make it the best choice for displaying some types of hidden information.

One other limitation was that the analysis technique was only applied to interactions where the errors were already known. A model of the perfect, or generic, interaction was needed to reveal the errors. Will the technique reveal previously unknown errors? If an analysis showed no problems then it maybe that the interaction was already perfect or that the designer’s model of it was not good enough. The technique is informal so there is no way to know if all the items of hidden information have been found. Further work is necessary to find out if this is a problem. New widgets, with no known problems, could be investigated to discover the effectiveness of the technique at exposing unknown errors.

The naïve psychological assumptions about where the user was looking were not precise. This meant that the designer had to make an educated guess about where the user was actually looking and this requires skill. The guess may be wrong and this could lead to the wrong information being made demanding or avoidable. An analysis of how a user interacts with a widget would have to performed to get some accurate data about where the user was really looking at any time.

Some types of auditory feedback are potentially annoying. In terms of the categorisation, demanding and sustained feedback is likely to be so. In the case of the modal dialogue box then this might be acceptable because it was marking an important mode in the system (perhaps an error that must be dealt with, for example). In most of the other cases this type of feedback had to be converted to action-dependent and transient to avoid annoyance. Graphics may be more appropriate in this case. The ‘fizzy’ windows described by Lee [105] effectively indicated the active window and were action-independent, demanding and sustained without being annoying. This shows one of the drawbacks of the analysis technique in that everything must be presented in sound. Further work is needed to find out more about the best types of information to present in sound.

8.3.3 Drawbacks with the structured method and its evaluation

The structured method has only been shown to work in two examples. Due to errors in data collection the third example did not show any improvements (although it was no worse). More widgets must be tested before the technique can really be said to be effective. Lee [105] has suggested that short experiments of the type that were used here were not good for investigating action slips. He says that longer term experiments are necessary so that subjects have time to become familiar with the experimental task they must perform. Unfortunately, there would not have been enough time to test three sonically-enhanced widgets in this way as part of this thesis. To overcome these drawbacks, the experimental tasks were kept very simple so that the subjects could learn and rapidly become used to them. Further longer-term experiments could be carried out where subjects came back and participated more than once to find out the long-term effects of sonic enhancement.

One criticism of the scrollbar experiment was that it turned out to be very difficult to generate many kangaroo type errors. It could be that as the subjects were experienced scrollbar users they had developed strategies for avoiding kangarooing in their everyday work which they used in the experiment. However, two subjects said that the sounds did help them identify when a kangaroo error had taken place. To overcome this, the time pressure of the experiment could be increased. The subjects would then make more errors.

The experiments only considered half of the problem of annoyance. They investigated whether the sounds were annoying for the primary user of the computer system but not others nearby. This is still an important gap in the knowledge of the use of sound at the interface. Another problem occurred with the measurement of annoyance. Subjects had to rate it on a scale as part of the workload analysis. The question they were asked was: "How annoying did you find the feedback from the task?" (see Appendix C Table 2). This description may have been too vague for subjects to be able to decide if the feedback was annoying or not. More precise questions could be asked, for example "Did you find the constant sound in the scrollbar annoying?". This would also allow a more detailed investigation of annoyance because each sound would be measured individually.

At the end of the chapter a future system was described where all of the widgets had integrated sounds. Each application would have its own timbre and spatial location. These would be used as the base for each of the widgets within an application. All of the widgets would work together as a whole (this is also described in more detail in the following section). The structured method does not support this integration. The widgets designed using it would not necessarily work together because each is designed separately. For example, the sonically-enhanced window suggested having a constant background tone indicating the active window. The sonically-enhanced button required a constant tone only when the mouse was pressed down over a screen button. These two could conflict. The interface designer would have to take all of the sonically-enhanced widgets and then re-design some of them so that they all fitted together. The structured method does not help with this.

8.4 FUTURE WORK

At the end of each of the work chapters future research issues pertaining to that chapter were discussed. This section will concentrate on the general issues that should be carried forward from this research. For the detailed descriptions the reader is referred to the chapter where the work was discussed.

8.4.1 A sonically-enhanced widget set

One area of future work would be to investigate more sonically enhanced widgets. In Chapter 6 several as yet un-tested improvements to standard graphical widgets were made. These were: Caps-lock keys, dialogue boxes, scrollbar dragging, button palettes and menus. These could be investigated in the same way as the widgets in Chapter 7. The same testing framework could be used measuring the same types of errors and workload.

The eventual aim of this aspect of the work would be to produce a complete toolkit of sonically-enhanced widgets that would be designed to work together as a whole. As discussed in the previous section the structured method would be used to investigate each widget in turn and this could lead to conflicts. At one time one widget might need silence to indicate it has finished and another might need a continuous sound. In Chapter 7 this was discussed and it was shown that conflicts could be avoided. For example, in the case of the scrollbar, its continuous tone must be made to fit with the continuous tone from the application as a whole. This could be done by merging the continuous tone from the scrollbar with the application (or window) sound. The sound would then change as the user scrolled through the document. This would also reduce the number of sounds necessary because the scrollbar sound and the application sound could be combined. The button earcons would work in the same way as the scrollbar ones. The mouse over a button sound would be played at a different pitch to the continuous window sound and slightly louder so that it stood out. The success sound would be mixed in with the continuous sounds, as in the scrollbar. A slip-off error could result in silence with all the sounds being turned off for a short time. These examples show that with simple modifications the sonically-enhanced widgets can be made to work together.

An interface designer would use these sonically-enhanced widgets in the same way as standard graphical ones when building an interface, increasing the usability. There are, however, many issues still to be investigated before this would be possible. For example, what control should users have over the sounds? Some might not like the timbres chosen or, due to hearing problems, not be able to hear high pitches used and so might need to change the sounds. Often the ability to customise a graphical interface is seen as a good thing. Users like to be able to change the colours on the screen, for example. If they are allowed to change the timbres, pitches, etc. then they make the sounds ineffective. It may be that not enough is known at the present time to allow users to change the sounds or they may render them ineffective.

8.4.2 Integrating sound into the whole user interface

The research described in this thesis integrated sound into graphical widgets such as scrollbars, buttons and windows. In a system designed in this way, the currently active application would make sound when its widgets were used. The next step forward for research in this area would be to extend the use of sound into a multiprocessing environment. In this type of situation all the applications would make sound, not just the currently active one. For example, in a desktop computer interface sounds would be produced by a wordprocessor when the user interacted with it as the active process. Whilst this is happening, sound might be produced by a compiler to indicate how it is progressing; a print job might indicate when it prints a page; and a complex spreadsheet recalculation might indicate when it has finished. As another example, in a process control environment many different parts of the plant might give continuous auditory feedback to the operator about their state. If he/she wanted to concentrate on one part of the system, the sounds of the other parts could be made to fade into the background. If something went wrong in another part of the system its sound could be made attention grabbing to warn the operator. The sounds would have to be carefully designed to work together so that they did not mask one another. Gaver [79] began to investigate some the problems associated with sound in this type of environment in the ARKola system (see Chapter 3).

The ESM technique could be extended from predicting where hidden information occurs in individual widgets to where it might occur in whole applications. For example, the feedback from a compiler running in the background is status information. It continues over time and shows the state of the compilation. It should be avoidable, action-independent, sustained and dynamic. If something happens, for example an error is produced, then this event should be signalled in a demanding way to grab the user’s attention.

Some of these ideas were initially tested in the sonically-enhanced window experiment in Chapter 7. In that experiment each window acted like a separate application (each having its own timbre and spatial location). This has shown that it is possible to use sound in this way. The work in Chapter 5 on presenting earcons in parallel could also be used. As each of the applications would be making sounds all the time, they would have to be designed not to conflict with each other. The guidelines from Chapter 5 will enable this. The work in this thesis has laid the foundations for this investigation and shown that is possible.

The system could be evaluated using the testing framework described in Chapter 7. An interface with sound could be compared to an interface with no sound. Time, errors and workload measures would again be used. Annoyance should be specifically tested to see if more sound in the interface makes it more annoying. Overall preference should also be used to obtain the users overall feelings towards the system.

The main aim of this research would be to evaluate the effectiveness of multiple simultaneous sounds in a human-computer interface. The work would show if using multiple sensory modalities could reduce the overload of any one modality, increase user performance or reduce workload. One other result would be to produce a set of guidelines, based on principles established by the investigation of multimodal interfaces. These would allow designers to add effective sounds to interfaces and permit the clear, consistent and effective use of sound across all applications in the interface (in a similar way to those described earlier in this thesis).

8.5 CONCLUSIONS AND CONTRIBUTIONS OF THE THESIS

The contributions made to the field of auditory interface design will now be described. Even with the limitations discussed above, the thesis is a major step forward. It proposes a structured method for integrating non-speech sound into human-computer interfaces. Prior to this work there was no method for doing this, sound had to be added in an *ad hoc* way by individual designers. This sometimes led to ineffective and inconsistent uses of sound. Adding sound using the structured method means that a designer can be certain that the sounds will improve usability. The sounds themselves will be effective and will have been added to reveal hidden information.

Before the research described here, much of the work in the field of auditory interface design had been to show that adding sound was possible. This research has taken the field a step further. It not only shows that adding sound to interface widgets is effective (and that multimodal interfaces are useful) but it also provides a method to allow a designer to do it simply across an interface. This work is some of the first to be formally experimentally evaluated. The advantages of sound have been demonstrated and experimentally proven.

The contribution of the thesis will now be discussed in terms of the two research questions and the structured method. Firstly, the research into what sounds to use will be discussed.

8.5.1 What sounds should be used at the interface?

The work on earcons is a major contribution to the area. Until this research they had never been tested. A detailed analysis of earcons showed that, if they were carefully constructed, high rates of recognition could be achieved. This was the most detailed analysis ever undertaken of a method for presenting information in sound. Interface designers can now be sure that earcons are an effective method of communicating information. The research showed that the rules suggested by Blattner *et al.* for creating earcons were too subtle and led to listeners not being able to differentiate them. Practical guidelines based on the results of the research were created. These could be used by an interface designer to create effective sounds.

Parallel earcons took the use of auditory feedback a stage further. One problem with combining earcons to make more complex messages is that they take longer to play.

This was overcome by playing the component parts in parallel. By introducing concepts of auditory stream segregation there was no reduction in recognition. Earcons can now keep pace with interactions.

All of these experiments investigated whether musical skill was necessary to be able to extract the structure from earcons. The results showed that it was not. This type of analysis had never before been undertaken. Knowing that non-musicians can recognise earcons means that they can be used in general interfaces for a wide variety of users. The research undertaken therefore covered a broad area. It has opened the way for the use of earcons in auditory interfaces of all types.

8.5.2 Where should sounds be used at the interface?

The analysis technique to find hidden information and make predictions about the type of feedback needed to make it explicit is a major advance. Prior to this research there was no way of doing this: Individual designers added sound in *ad hoc* ways. If the research area of auditory interfaces was to grow then a more principled approach had to be taken. This thesis suggested that if sound was used to display hidden information then usability could be improved. Using the technique some errors with basic interface widgets were identified and suggestions made for their improvement. The technique was easy to use and provided predictions in a form that were easy to understand and apply.

The technique showed that using more visual feedback to overcome problems of hidden information would not necessarily work. Visual feedback is, by its nature, avoidable: A user might choose not to look at it and this can cause problems. For example, with screen buttons the user was no longer looking at the button when it gave error information. Due to closure, attention had moved on to the next interaction and the previous one was outside the area of visual focus. Sound can overcome this because it is omni-directional. The technique also takes account of possible annoyance due to sound (such as with continuous demanding feedback) and can minimise it.

8.5.3 The structured method

The analysis technique was combined with the earcon guidelines to produce the structured method for integrating sound into user interfaces. If a designer followed the series of steps in the method then he/she could be sure that effective sounds had been added that improved usability. This is a very important step forward. The results showed that making hidden information explicit can improve usability.

Sonically-enhanced scrollbars and buttons were shown to improve usability. These were practical examples of the structured method in action. Widgets that combined both auditory and visual feedback were more effective as they made use of the natural way that humans deal with information in everyday life. The graphical versions of these widgets occur in almost all graphical interfaces. These enhanced ones could simply be added and the interfaces made more usable.

In the button experiment it was shown that sound could be used to replace the visual feedback present in the standard button. This leads the way to removing other graphical feedback and replacing it with more effective auditory feedback, reducing the complexity of the graphical display and leaving the visual system to concentrate on the main task the user is trying to accomplish.

This thesis undertook the first formal experimental analysis of the potential annoyance of sound at the interface. The results showed that in the three experiments none of the sonically-enhanced widgets were more annoying than their visual counterparts. Both continuous and discrete sounds of different pitches and intensities have been tested and showed no significant differences in annoyance. This was because the sounds used provided information that the user needed, they were not gimmicks. This counters many of the claims that auditory interfaces would be annoying to use.

One of the strengths of the work described is that it has all been experimentally evaluated, putting the thesis on strong foundations. Both earcons and the analysis technique were tested and shown to be effective. Future auditory interface designers can use the structured method safe in the knowledge that it works.

In conclusion, the work in this thesis aimed to answer two questions: Where to use sound at the interface and what sounds to use at the interface. By answering these questions a structured method was produced. All aspects of it have been evaluated to make sure it has strong foundations. This method will make the development of multimodal interfaces much simpler. A designer not skilled in sound design can now create such interfaces that will improve usability but not be annoying. This means that sound can now be used in more interfaces and the advantages of the combination of auditory and visual information can help in the everyday interactions of users with their computers.

APPENDIX A: RAW DATA FROM EARCON EXPERIMENTS 1 & 2

A.1 INTRODUCTION

This appendix contains the raw data referred to in Chapter 4. Data is provided for the musical, simple, control and new groups for Experiments 1 & 2. It also contains data for the three earcon compounds and musicians versus non-musicians.

Musical Group		Phase I			Phase II		Phase III			Phase IV				
Subjects	Musician	Type	Family	File	Menu	Item	Type	Family	File	Type	Family	File	Menu	Item
s1	n	5	4	0	6	4	6	4	0	4	5	0	8	6
s2	y	6	8	3	9	6	7	9	1	7	8	2	10	7
s3	y	8	9	1	9	9	6	5	0	7	8	1	11	10
s4	y	8	6	2	9	10	9	2	1	8	1	3	9	9
s5	y	2	3	0	9	7	3	4	0	5	1	2	13	10
s6	y	3	9	1	9	8	2	5	0	3	6	0	9	8
s7	n	5	9	3	9	9	7	10	1	4	7	1	10	9
s8	n	5	9	0	8	2	6	8	1	5	10	0	6	4
s9	n	4	9	1	8	5	6	9	2	2	4	1	6	3
s10	n	5	9	1	9	9	6	10	3	9	8	0	11	10
s11	n	3	2	1	9	7	1	1	0	0	0	0	5	4
s12	y	5	9	1	9	9	7	9	1	10	9	2	7	9
		out of 10	out of 10	out of 3	out of 10	out of 10	out of 10	out of 10	out of 3	out of 12	out of 12	out of 3	out of 13	out of 13
Simple Group		Phase I			Phase II		Phase III			Phase IV				
Subjects	Musician	Type	Family	File	Menu	Item	Type	Family	File	Type	Family	File	Menu	Item
s1	n	6	3	0	4	3	4	2	0	6	0	0	1	1
s2	y	5	4	1	6	6	4	5	1	6	4	2	8	8
s3	y	9	5	2	8	10	10	8	3	11	7	1	9	10
s4	n	8	4	1	8	10	10	5	3	8	6	3	9	9
s5	n	4	1	1	9	9	1	0	0	1	0	0	10	11
s6	n	4	2	2	7	9	2	0	0	4	4	0	6	8
s7	n	3	3	2	9	6	4	3	1	2	3	1	6	2
s8	y	9	5	0	6	6	10	3	0	7	6	0	3	1
s9	y	9	5	3	9	10	9	7	2	10	6	1	9	10
s10	n	4	3	2	8	6	4	3	1	4	2	0	4	3
s11	y	8	5	2	8	10	10	7	2	9	5	0	10	12
s12	n	4	6	1	9	5	4	5	2	6	3	1	8	5
		out of 10	out of 10	out of 3	out of 10	out of 10	out of 10	out of 10	out of 3	out of 12	out of 12	out of 3	out of 13	out of 13
Control Group		Phase I			Phase II		Phase III			Phase IV				
Subjects	Musician	Type	Family	File	Menu	Item	Type	Family	File	Type	Family	File	Menu	Item
s1	n	4	7	1	6	3	1	9	0	1	3	0	3	2
s2	n	4	9	0	7	5	3	5	1	2	2	0	5	4
s3	n	2	6	1	9	3	1	5	0	1	4	0	8	2
s4	y	3	2	1	9	3	2	2	1	0	3	0	10	4
s5	y	7	4	2	9	6	5	4	0	4	5	1	12	10
s6	y	5	9	0	10	6	3	7	0	5	8	2	12	8
s7	n	4	8	0	8	4	2	7	1	4	7	0	6	1
s8	n	7	8	0	8	3	5	6	1	1	6	0	7	2
s9	y	7	10	1	9	2	7	10	1	7	9	1	7	2
s10	y	5	9	0	8	5	5	9	2	6	11	1	9	4
s11	y	7	9	0	6	5	8	6	1	1	1	0	5	5
s12	n	6	9	0	8	3	8	10	3	8	11	2	6	3
		out of 10	out of 10	out of 3	out of 10	out of 10	out of 10	out of 10	out of 3	out of 12	out of 12	out of 3	out of 13	out of 13

***Appendix A Table 1**: Data for Musical, Simple and Control Groups in Experiment 1. It includes data for musicians versus non-musicians.*

New Group		Phase I			Phase II		Phase III			Phase IV				
Subjects	Musician	Type	Family	File	Menu	Item	Type	Family	File	Type	Family	File	Menu	Item
s1	y	10	9	3	10	9	10	10	3	12	10	2	12	10
s2	n	9	9	3	10	10	9	9	3	10	7	3	7	8
s3	y	5	7	0	10	8	1	6	0	5	9	1	12	12
s4	n	9	4	3	9	3	6	4	2	7	6	0	6	1
s5	y	10	9	3	8	7	10	10	3	12	12	3	10	8
s6	n	9	6	3	5	3	10	7	3	11	7	1	3	1
s7	n	9	6	1	10	6	8	6	0	6	5	0	10	7
s8	y	6	5	2	8	6	5	5	1	3	7	0	7	7
s9	n	10	9	3	10	9	10	10	3	12	9	3	11	11
s10	n	7	8	0	7	7	10	9	0	8	6	0	7	6
s11	y	10	10	2	9	9	10	9	3	12	10	2	13	12
s12	y	10	10	3	8	10	10	9	3	12	10	3	11	13
		out of 10	out of 10	out of 3	out of 10	out of 10	out of 10	out of 10	out of 3	out of 12	out of 12	out of 3	out of 13	out of 13

Appendix A Table 2: *Data for New Group in Experiment 2. It includes data for musicians versus non-musicians.*

Musical Group								
Subjects	Menu	Item	Menu	Item	Family	Type	File	Total
s1	0	0	1	0	1	1	0	3
s2	0	0	0	0	1	1	0	2
s3	1	1	1	1	1	1	1	7
s4	0	0	1	1	0	1	1	4
s5	1	1	1	1	0	1	0	5
s6	1	0	1	1	0	0	0	3
s7	1	0	1	1	0	0	0	3
s8	0	0	1	1	1	0	0	3
s9	1	1	0	0	0	0	0	2
s10	1	1	1	1	0	1	0	5
s11	1	1	1	1	0	0	0	4
s12	0	0	1	1	1	1	0	4
Simple Group								
Subjects	Menu	Item	Menu	Item	Family	Type	File	Total
s1	0	0	0	0	0	0	0	0
s2	1	1	1	1	1	1	1	7
s3	1	1	1	1	1	0	0	5
s4	0	0	1	1	1	1	0	4
s5	1	1	0	1	0	0	0	3
s6	0	0	1	1	1	0	0	3
s7	0	0	1	1	1	1	1	5
s8	0	0	0	0	1	1	0	2
s9	0	0	1	1	0	0	0	2
s10	1	1	0	0	1	1	0	4
s11	1	1	1	1	0	1	0	5
s12	1	0	1	1	0	1	0	4
Control Group								
Subjects	Menu	Item	Menu	Item	Family	Type	File	Total
s1	1	1	0	0	0	0	0	2
s2	1	1	0	0	0	0	0	2
s3	0	0	1	1	0	0	0	2
s4	1	0	0	0	0	0	0	1
s5	1	1	1	1	0	0	0	4
s6	1	1	1	1	1	0	0	5
s7	1	0	1	0	1	0	0	3
s8	1	1	0	0	0	0	0	2
s9	1	1	1	0	1	1	1	6
s10	1	1	1	0	0	0	0	3
s11	1	1	0	0	0	0	0	2
s12	1	1	0	0	1	0	0	3

Appendix A Table 3*: Data for three earcon compounds from Experiment 1*

APPENDIX B: RAW DATA FROM THE PARALLEL EARCONS EXPERIMENT

B.1 INTRODUCTION

This appendix contains the raw data referred to in Chapter 5. Data is provided for the serial and parallel groups for each of the phases. It also contains data for musicians versus non-musicians.

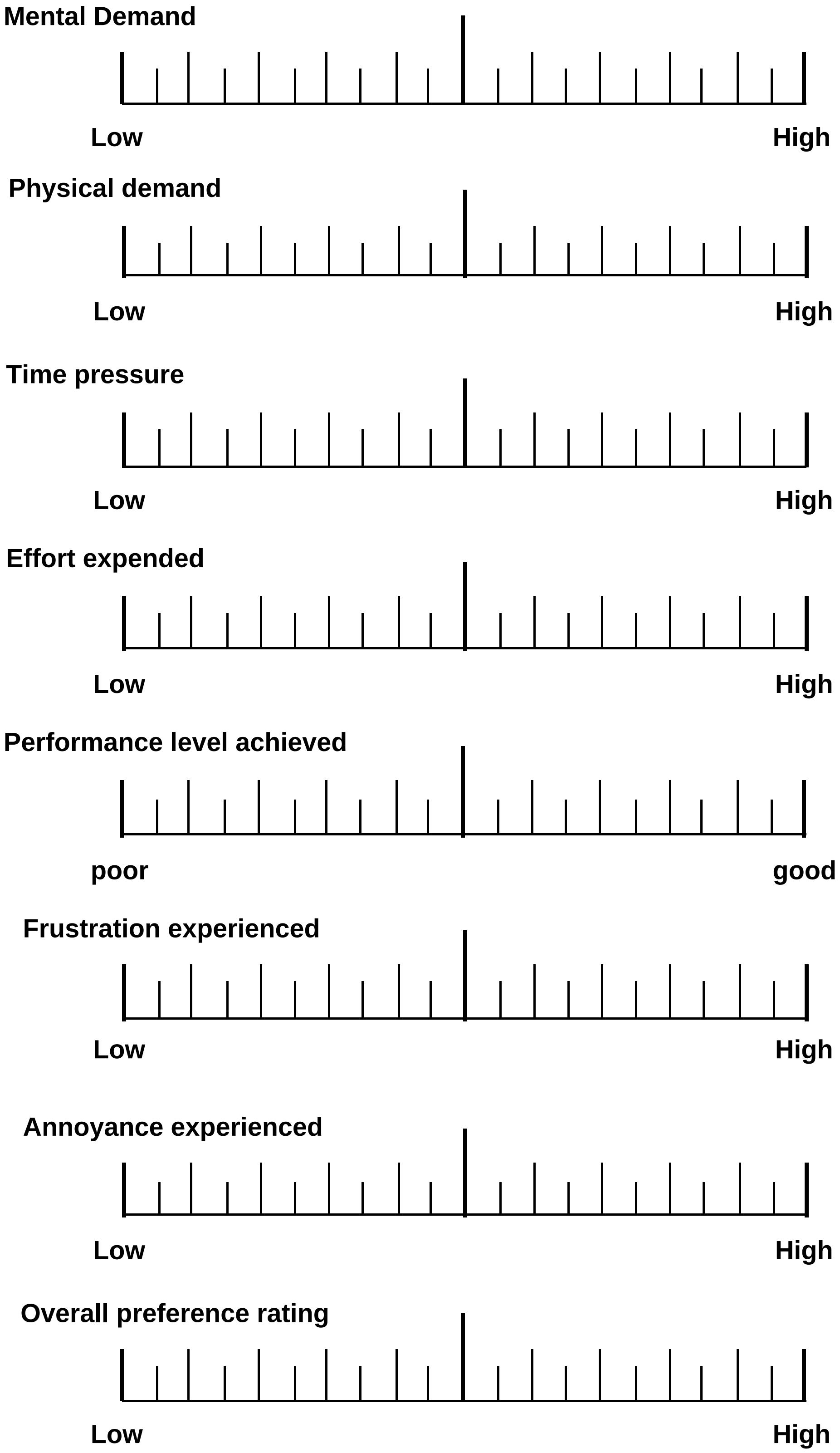
Serial Group		Phase I		Phase II		Phase III(1)				Phase III(2)			
Subjects	Musician	Type	Family	Menu	Item	Type	Family	Menu	Item	Type	Family	Menu	Item
s1	n	8	6	9	7	7	4	9	4	7	7	9	5
s2	y	8	8	8	6	7	9	8	5	7	9	8	5
s3	y	8	9	9	7	8	9	7	6	9	9	8	7
s4	n	6	8	9	7	2	1	8	3	2	3	8	3
s5	y	9	8	9	6	7	4	9	4	7	7	9	5
s6	n	7	9	9	9	8	8	9	8	9	8	9	8
s7	n	9	8	9	8	8	7	9	6	9	7	9	8
s8	y	9	9	9	9	7	9	9	8	9	9	9	9
s9	y	8	9	9	8	8	9	9	8	9	9	9	9
s10	n	6	8	9	9	1	4	9	6	2	6	9	7
s11	n	8	9	9	9	8	8	9	9	8	9	9	9
s12	y	9	9	9	9	9	9	8	9	9	9	9	9
		out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9
Parallel Group		Phase I		Phase II		Phase III(1)				Phase III(2)			
Subjects	Musician	Type	Family	Menu	Item	Type	Family	Menu	Item	Type	Family	Menu	Item
s1	n	9	9	9	9	9	8	8	7	9	8	9	8
s2	n	9	6	9	9	3	9	9	8	4	9	9	9
s3	y	9	9	9	9	9	7	8	5	9	8	8	7
s4	n	9	9	9	9	5	7	8	7	8	8	8	8
s5	n	9	9	9	9	8	9	7	7	9	9	9	9
s6	y	9	9	9	8	8	7	9	6	9	8	9	9
s7	y	9	8	9	9	8	8	8	5	8	9	9	8
s8	n	6	6	9	8	6	3	7	5	8	3	8	7
s9	y	6	6	9	9	8	9	9	8	8	9	9	8
s10	y	9	8	9	8	7	8	7	1	7	8	8	5
s11	n	9	9	9	9	9	6	8	4	9	9	8	8
s12	y	8	9	9	7	8	6	8	3	8	7	9	5
		out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9	out of 9
Rejected Subjects		Phase I		Phase II		Phase III(1)				Phase III(2)			
Subjects	Musician	Type	Family	Menu	Item	Type	Family	Menu	Item	Type	Family	Menu	Item
s1	n	6	6	7	4	6	4	8	2	8	5	8	2
s2	n	4	9	9	4	1	8	6	2	3	9	6	2
s3	n	9	9	9	5	9	8	8	5	9	9	9	5

Appendix B Table 1: *Data for Serial and Parallel Groups in the parallel earcons experiment. It includes data for musicians versus non-musicians and rejected subjects. The rejected subjects were all from the Serial Group.*

APPENDIX C: RAW DATA FOR THE SONICALLY-ENHANCED WIDGETS EXPERIMENTS

C.1 INTRODUCTION

This appendix contains the raw data from the sonically-enhanced widgets experiments described in Chapter 7. It also contains tables and descriptions of workload tests given to subjects in the experiments.



***Appendix C Table 1:** Workload charts used in the audio-enhanced widgets experiments.*

Rating scale definitions

Title	endpoints	Description
mental demand	Low/High	How much mental, visual and auditory activity was required? (e.g. thinking, deciding calculating, looking, listening, cross-monitoring, scanning, searching)
physical demand	Low/High	How much physical activity was required? (e.g. pushing, pulling, turning, controlling)
time pressure	Low/High	How much time pressure did you feel because of the rate at which things occurred? (e.g. slow, leisurely, rapid, frantic)
effort expended	Low/High	How hard did you work (mentally and physically) to accomplish your level of performance?
performance level achieved	Poor/Good	How successful do you think you were in accomplishing the mission goals?
frustration experienced	Low/High	How much frustration did you experience? (e.g., stress, irritation, discouragement)
annoyance experienced	Low/High	How annoying did you find the feedback from the task ?
overall preference	Low/High	Rate your preference of the two scrollbars. Which one made the task the easiest?

Appendix C Table 2: *Workload descriptions given to subjects when filling in the workload charts. The descriptions are taken from* [122].

C.2 SCROLLBAR EXPERIMENT RAW DATA

This section contains the raw data for the scrollbar experiment. First there is the raw workload data and this is followed by time and error data.

Subject	Condition	Mental	Physical	Time	Effort	Annoyance	Frustration	Performance	Overall
s1	a	12	6	10	10	12	8	10	16
	v	16	6	10	14	14	13	8	4
s2	a	17	9	4	10	16	10	18	4
	v	18	9	5	13	16	12	16	2
s3	a	17	2	6	17	17	18	13	6
	v	17	2	6	16	6	14	10	14
s4	a	16	3	9	13	12	14	4	13
	v	17	1	9	16	9	9	9	7
s5	a	11	11	7	10	11	11	11	15
	v	15	15	10	7	15	5	6	5
s6	a	12	1	5	12	8	9	11	12
	v	10	1	4	14	11	6	10	8
s7	a	10	5	9	6	9	9	8	16
	v	17	4	14	13	17	17	3	0
s8	a	9	6	4	7	2	3	13	13
	v	12	6	4	10	1	4	12	8
s9	a	20	20	0	16	0	8	0	16
	v	20	20	0	20	0	0	4	10
s10	a	16	9	12	17	6	5	7	11
	v	19	9	12	19	6	12	9	6
s11	a	14	1	3	16	6	8	17	14
	v	16	1	3	16	0	4	19	6
s12	a	8	3	3	10	12	10	3	4
	v	12	3	3	6	7	6	8	10
Totals	Auditory (total)	162	76	72	144	111	113	115	140
	Auditory (average)	13.5	6.3	6	12	9.2	9.4	9.5	11.6
	Visual (total)	189	77	80	164	102	102	114	80
	Visual (average)	15.7	6.4	6.6	13.6	8.5	8.5	9.5	6.6

***Appendix C Table 3**: Workload data for the scrollbar experiment.*

Subject	Search Tasks				Navigate Tasks			
s1	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2623			1	1397		
	2	2286			2	1170		
	3	4619			3	1891		
					4	979	1	
					5	573		
					6	1012	1	
					7	2257		
					8	690		
					9	3243	2	3
					10	1066		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1895			1	1167		
	2	1963			2	1109		
	3	3565	1		3	1155		
					4	2767	1	
					5	681		
					6	489		
					7	1400		
					8	518		
					9	760		
					10	853		
s2	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2144			1	3305	1	
	2	1015			2	947		
	3	5872			3	1284		
					4	374		
					5	445		
					6	540		
					7	1736		
					8	597		
					9	1235		
					10	1083		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1387			1	1280		
	2	1514	1		2	902		
	3	6843			3	2132	2	
					4	522		
					5	478		
					6	492		
					7	1286		
					8	571		
					9	1260		
					10	611		
s3	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2753			1	1018		
	2	2097			2	1147		
	3	2681			3	1752	1	
					4	582		
					5	1107	1	
					6	1591	1	
					7	1025		
					8	560		
					9	1161		
					10	3978	3	
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1730			1	886		
	2	1700	1		2	1089	1	
	3	1737			3	1178		
					4	480		
					5	502		
					6	779		
					7	1170		
					8	569		
					9	974		
					10	980		

Appendix C Table 4: *Raw data for scrollbar experiment.*

s4	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2057	1		1	3344	2	
	2	1001			2	925		
	3	2308	1		3	1271		1
					4	1454	1	
					5	496		
					6	2400	2	
					7	2000		
					8	708		
					9	1158		
					10	776		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1497	1		1	2158		3
	2	1124	1		2	821		
	3	2276			3	1738	1	
					4	870	1	
					5	772		
					6	645		
					7	1719		
					8	1411	1	
					9	3180	2	
					10	758		
s5	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	4792	1		1	6079	2	
	2	2179			2	12523	2	1
	3	11150	1		3	4154	2	
					4	1415	1	
					5	1984	1	
					6	868		
					7	2605		
					8	920		
					9	2026		
					10	1353		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	3917			1	2011	1	
	2	1811			2	943		
	3	7659			3	2913	2	
					4	677		
					5	510		
					6	614		
					7	1626		
					8	703		
					9	1028		
					10	1512		
s6	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1652			1	856		1
	2	1409	1		2	776		
	3	3527			3	1558		
					4	739		
					5	496		
					6	558		
					7	999		
					8	548		
					9	3577	2	1
					10	2037	1	
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1422	1		1	939		
	2	1306			2	625		
	3	2522			3	1368	1	
					4	470		
					5	568		
					6	452		
					7	1373		
					8	675		
					9	785		
					10	912	1	
s7	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	3100			1	2270		
	2	3195		1	2	3030	1	

Appendix C Table 4: *Raw data for scrollbar experiment.*

	3	26073	1	2	3	7220	2	
					4	838		
					5	940		
					6	1309	1	
					7	7118	3	1
					8	5310	2	
					9	2117		
					10	1404	1	
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2609	1		1	1541		
	2	3504			2	1074		
	3	4619	1		3	2598	2	
					4	695		
					5	1372	1	
					6	805		
					7	1687		
					8	1371		
					9	2262		
					10	982		
s8	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2461	1		1	1077		
	2	2109			2	1194		
	3	6361			3	1236		
					4	432		
					5	473		
					6	619		
					7	1591		
					8	549		
					9	1053		
					10	739		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2142			1	1506	1	
	2	3347		3	2	898		
	3	6483			3	1158		
					4	429		
					5	929	1	
					6	494		
					7	1318		1
					8	508		
					9	865		
					10	705		
s9	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2169	1		1	1693		
	2	2659	1		2	1315		
	3	11311			3	2088		
					4	1033		
					5	945		
					6	953		
					7	2168		
					8	2526	1	
					9	1491		
					10	1472		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2115	1		1	1447		
	2	1287	1		2	1068		
	3	4532	1		3	4895	3	
					4	906		
					5	1705	1	
					6	768		
					7	1753		
					8	2080	2	
					9	2119	1	
					10	1065		
s10	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2361			1	2019		
	2	5174			2	6793	4	
	3	2825			3	6429	5	
					4	796		

Appendix C Table 4: *Raw data for scrollbar experiment.*

					5	628		
					6	3551	2	
					7	1505		
					8	758		
					9	1725		
					10	977		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1430	1		1	1780	1	
	2	1520			2	839		
	3	1869			3	5626	4	
					4	950	1	
					5	664		
					6	642		
					7	1698		
					8	1542	1	
					9	1196		
					10	937		
s11	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	3176			1	1049		
	2	2450			2	936		
	3	5440			3	1355		
					4	522		
					5	630		
					6	698		
					7	2160		
					8	631		
					9	1081		
					10	766		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	1836			1	1120		
	2	3612	1		2	789		
	3	3243		1	3	2490	2	
					4	508		
					5	410		
					6	367		
					7	1321		
					8	615		
					9	847		
					10	544		
s12	VISUAL							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2345	1		1	4984	2	
	2	3789			2	819		
	3	5070	1		3	1069		
					4	600		
					5	648		
					6	738		
					7	1146		
					8	679		
					9	1433		
					10	684		
	AUDITORY							
	search task	time (60ths)	wrong page	kangaroos	navigate task	time (60ths)	wrong page	kangaroos
	1	2975		1	1	1377		
	2	5285			2	1051		
	3	7657			3	5029	3	
					4	494		
					5	1198	1	
					6	639		
					7	1442		
					8	620		
					9	1570		
					10	1827	1	

Appendix C Table 4: *Raw data for scrollbar experiment.*

C.3 BUTTON EXPERIMENT RAW DATA

This section contains the raw data for the button experiment. First the raw workload data is presented and then the overall data for the different types of errors.

Subject	Condition	Mental	Physical	Time	Effort	Annoyance	Frustration	Performance	Overall
s1	a	4	14	10	17	17	11	10	13
	v	4	14	10	18	15	15	10	7
s2	a	6	15	10	14	5	16	10	16
	v	4	15	7	10	5	11	13	7
s3	a	13	13	14	7	14	13	13	12
	v	15	14	15	7	15	14	13	11
s4	a	2	16	0	9	0	1	19	13
	v	4	16	0	10	0	0	19	10
s5	a	7	17	2	5	3	1	17	14
	v	14	17	5	11	2	2	17	11
s6	a	13	10	10	12	13	14	10	15
	v	18	11	10	17	18	18	10	8
s7	a	5	15	1	13	2	2	14	10
	v	3	15	1	11	1	2	13	10
s8	a	3	18	0	17	4	16	17	4
	v	2	18	0	16	1	16	19	2
s9	a	4	3	4	17	0	0	13	15
	v	9	3	7	18	0	4	13	11
s10	a	2	11	1	8	3	7	11	12
	v	4	11	3	9	4	7	9	8
s11	a	6	3	10	15	11	10	19	4
	v	7	3	10	14	15	13	19	1
s12	a	17	16	4	13	11	14	16	14
	v	14	12	1	9	16	13	15	11
Totals	Auditory (total)	82	151	66	147	83	105	169	142
	Auditory (average)	6.83	12.58	5.5	12.25	6.91	8.75	14.08	11.83
	Visual (total)	98	149	69	150	92	115	170	97
	Visual (average)	8.16	12.41	5.75	12.5	7.66	9.58	14.16	8.08

***Appendix C Table 5**: Workload data for the button experiment.*

Subjects	Condition	Total slip-offs	Total background clicks	Total codes typed
s1	a	6	54	51
	v	1	16	50
s2	a	0	43	58
	v	0	37	67
s3	a	2	40	67
	v	1	66	74
s4	a	19	14	68
	v	6	12	61
s5	a	0	8	56
	v	0	9	62
s6	a	5	18	52
	v	0	18	49
s7	a	1	32	66
	v	0	40	64
s8	a	3	23	79
	v	0	17	78
s9	a	24	61	86
	v	8	20	77
s10	a	0	30	56
	v	2	20	77
s11	a	19	203	79
	v	11	64	70
s12	a	0	3	56
	v	6	3	57

***Appendix C Table 6**: Overall data for the button experiment.*

Subjects	Auditory condition			Visual condition		
	slip-off (60ths sec.)	recover (60ths sec.)	mouse ups and downs	slip-off (60ths sec.)	recover (60ths sec.)	mouse ups and downs
s1	432474	432626	1	361388	361691	5
	446861	447002	1			
	449451	449535	1			
	462584	462648	1			
	467683	467704	1			
	469075	469194	1			
s2	no slip offs			no slip offs		
s3	158309	158473	1	236096	236196	3
	160853	160940	1			
s4	899719	899852	3	800872	800914	1
	900831	900942	3	808841	810045	31
	903669	903784	1	816715	817119	9
	904875	905097	1	817899	818032	3
	905722	905797	1	835488	835840	9
	908702	908786	1	836421	836575	3
	915098	915195	1			
	916286	916622	9			
	919508	919594	1			
	919724	919802	1			
	921048	921124	1			
	923255	923309	1			
	923390	923475	1			
	925066	925647	17			
	926653	926790	1			
	928190	928248	1			
	928985	929105	3			
	929154	929261	1			
	930588	930675	1			
s5	no slip offs			no slip offs		
s6	343906	344108	3	no slip offs		
	359133	359166	1			
	369798	370154	5			
	375152	375338	3			
	389075	389313	3			
s7	1219545	1219695	1	no slip offs		
s8	868759	868864	3	no slip offs		
	870238	870333	1			
	874617	874691	1			
s9	761382	761483	1	691354	691484	3
	762124	762188	1	703583	704012	13
	762535	762615	1	706226	706299	1
	769927	770122	3	718693	718797	1
	771550	771632	1	719718	719811	1
	772440	772540	1	722028	722574	17
	773640	773718	1	724240	724330	3
	773907	773985	1	726934	727598	27
	775750	775834	1			
	776193	776275	1			
	777603	777688	1			
	778430	778495	1			
	779271	779363	1			
	783553	783599	1			
	786159	786220	1			
	786758	786815	1			
	790366	790441	1			
	790483	790575	1			
	800790	800869	1			
	801729	801873	1			
	802132	802213	1			
	805261	805332	1			
	805851	805929	1			
	806028	806107	1			
s10	no slip offs			155559	155817	3
				164702	164848	3
s11	296375	296428	1	241277	241397	3
	296485	296540	1	243519	243583	3
	299397	299478	1	253640	253743	3
	300072	300122	1	259762	259873	3

***Appendix C Table 7**: Raw data for slip-offs including time slip-off occurred, time until recovery and the number of mouse button ups and downs required for recovery in the button experiment.*

	301238	301309	1	260287	260385	1
	306453	306530	1	260577	260947	9
	306537	306590	1	269818	270351	19
	312198	312237	1	272535	272693	3
	312991	313084	1	272933	273442	12
	322940	323011	1	273122	273850	16
	326345	326423	1	273559	273672	3
	331228	331315	1			
	331735	331806	1			
	338360	338422	1			
	338617	338676	1			
	338775	338826	1			
	346145	346205	1			
	348565	348731	1			
	349505	349573	1			
s12	no slip offs			no slip offs		

Appendix C Table 7*: Raw data for slip-offs including time slip-off occurred, time until recovery and the number of mouse button ups and downs required for recovery in the button experiment.*

C.4 WINDOW EXPERIMENT RAW DATA

This section contains the raw data from the window experiment. The raw workload data is given first and the overall error results.

Subject	Condition	Mental	Physical	Time	Effort	Annoyance	Frustration	Performance	Overall
s1	a	12	2	15	11	2	13	8	15
	v	13	2	15	12	8	13	8	10
s2	a	18	4	10	17	17	6	16	6
	v	18	4	10	15	14	6	17	8
s3	a	18	10	4	12	2	19	20	8
	v	19	10	4	14	3	20	20	4
s4	a	4	8	2	20	18	6	19	10
	v	4	8	2	20	16	4	19	12
s5	a	14	14	12	16	14	10	8	11
	v	14	14	12	16	12	10	10	9
s6	a	10	11	13	11	13	15	9	8
	v	5	11	11	11	8	11	11	11
s7	a	10	12	15	15	5	17	12	20
	v	12	12	17	17	8	17	8	10
s8	a	8	10	10	10	12	10	12	10
	v	12	14	14	18	12	12	10	10
s9	a	15	6	9	16	14	17	15	8
	v	15	6	9	15	11	17	13	10
s10	a	16	10	14	0	0	0	20	10
	v	16	10	14	0	0	0	20	10
s11	a	8	14	0	12	10	20	18	2
	v	8	14	0	12	0	18	18	2
s12	a	16	13	13	17	17	17	9	11
	v	18	13	11	15	14	17	7	8
Totals	Auditory (total)	149	114	117	157	124	150	166	119
	Auditory (average)	12.41	9.5	9.75	13.08	10.33	12.5	13.83	9.91
	Visual (total)	154	118	119	165	106	145	161	104
	Visual (average)	12.83	9.83	9.91	13.75	8.83	12.08	13.41	8.66

***Appendix C Table 8**: Workload data for the window experiment.*

Subjects	Condition	References window errors	Total references window errors	Control window errors	Time to complete task (60ths sec.)
s1	a	0	0	0	1477
	v	3	9	0	1330
s2	a	1	4	3	1333
	v	3	8	0	1517
s3	a	8	22	0	1583
	v	7	20	2	1302
s4	a	2	4	2	2000
	v	9	25	2	2092
s5	a	7	13	3	1536
	v	3	11	1	1255
s6	a	1	2	1	1802
	v	3	7	1	1746
s7	a	0	0	1	1243
	v	0	0	1	1083
s8	a	0	0	1	1693
	v	0	0	2	1543
s9	a	2	3	1	1389
	v	6	8	1	1620
s10	a	2	9	0	1954
	v	0	0	2	1836
s11	a	0	0	1	1407
	v	2	4	0	1434
s12	a	0	0	0	1479
	v	0	0	0	1369

Appendix C Table 9*: Overall data for the window experiment.*

REFERENCES

1. Aldrich, F.K. & Parkin, A.J. (1989). Listening at speed. *British journal of visual impairment and blindness*, 7(1), pp. 16-18.

2. Allan, J.J. & Chiu, A.M. (1977). An effectiveness study of a CAD system augmented by audio feedback. *Computers & Graphics*, 2, pp. 321-323.

3. Alty, J. (1991). Multimedia-What is it and how do we exploit it? In D. Diaper & N. Hammond (Eds.), *Proceedings of HCI'91,* Edinburgh: Cambridge University Press, pp. 31-44.

4. Alty, J.L. & McCartney, C.D.C. (1991). Design of a multi-media presentation system for a process control environment. In *Eurographics multimedia workshop, Session 8: Systems,* Stockholm.

5. American National Standards Institute (1973). *American National Psychoacoustic Terminology* (No. S3.20). American National Standards Institute, New York.

6. American Standards Association (1960). *Acoustical Terminology* (No. S1.1). American Standards Association, New York.

7. Apple Computer Inc. (1985). Chapter Two: The Macintosh User-Interface Guidelines. In *Inside Macintosh: Volume I*, pp. 23-70. Reading, Massachusetts: Addison-Wesley.

8. Apple Computer Inc. (1991). Chapter Two: User Interface Guidelines. In *Inside Macintosh: Volume IV*, pp. 2-3 - 2-37. Reading, Massachusetts: Addison-Wesley.

9. Arons, B. (1992). A review of the cocktail party effect. *Journal of the American Voice I/O Society*, 12(July).

10. Avons, S.E., Leiser, R.G. & Carr, D.J. (1989). Paralanguage and human-computer interaction. Part 1: Identification of recorded vocal segregates. *Behaviour and Information Technology*, 8(1), pp. 21-31.

11. Badeley, A. (1990). *Human Memory: Theory and Practice*. London: Lawrence Erlbaum Associates.

12. Baecker, R., Small, I. & Mander, R. (1991). Bringing icons to life. In *Proceedings of CHI'91*, New Orleans: ACM Press, Addison-Wesley, pp. 1-6.

13. Ballas, J.A. & Howard, J.H. (1987). Interpreting the language of environmental sounds. *Envirionment and Behaviour*, 19(1), pp. 91-114.

14. Barfield, W., Rosenberg, C. & Levasseur, G. (1991). The use of icons, earcons and commands in the design of an online hierarchical menu. *IEEE Transactions on Professional Communication*, 34(2), pp. 101-108.

15. Barker, P.G. & Manji, K.A. (1989). Pictorial dialogue methods. *International Journal of Man-Machine Studies*, 31, pp. 323-347.

16. Begault, D.R. & Wenzel, E.M. (1990). *Techniques and applications for binaural sound manipulation in human-computer interfaces* (NASA Technical Memorandum No. 102279). NASA Ames Research Centre: California.

17. Berger, K.W. (1963). Some factors in the recognition in timbre. *Journal of the Acoustical Society of America*, 36(10), pp. 1888-1891.

18. Berglund, B., Preis, A. & Rankin, K. (1990). Relationship between loudness and annoyance for ten community sounds. *Environment International*, 16, pp. 523-531.

19. Bevan, N. & Macleod, M. (1994). Usability measurement in context. *International Journal of Man-Machine Studies*, 13(1 & 2), pp. 123-145.

20. Björk, E.A. (1985). The perceived quality of natural sounds. *Acustica*, 57(3), pp. 185-188.

21. Blandford, A., Harrison, M. & Barnard, P. (1994). Understanding the properties of interaction. *Amodeus 2 ESPRIT basic research action 7044*, Project working paper.

22. Blattner, M. & Dannenberg, R.B. (1992). Introduction: The trend toward multimedia interfaces. In M. Blattner & R. B. Dannenberg (Eds.), *Multimedia Interface Design*, pp. xvii-xxv. New York: ACM Press, Addison-Wesley.

23. Blattner, M. & Dannenberg, R.B. (Eds.). (1992). *Multimedia Interface Design*. New York: ACM Press, Addison-Wesley.

24. Blattner, M., Greenberg, R.M. & Kamegai, M. (1992). Listening to turbulence: An example of scientific audiolization. In M. Blattner & R. B. Dannenberg (Eds.), *Multimedia Interface Design*, pp. 87-104. New York: ACM Press, Addison-Wesley.

25. Blattner, M., Sumikawa, D. & Greenberg, R. (1989). Earcons and icons: Their structure and common design principles. *Human Computer Interaction*, 4(1), pp. 11-44.

26. Blattner, M., Papp, A. & Glinert, E. (1992). Sonic enhancements of two-dimensional graphic displays. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display*, Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 447-470.

27. Bly, S. (1982). *Sound and computer information presentation* (Unpublished PhD Thesis No. UCRL53282). Lawrence Livermore National Laboratory.

28. Bramwell, C.J. & Harrision, M.D. (1994). Design questions about interactive computer programs. *Unpublished paper*.

29. Bregman, A. (1992). Foreword. In G. Kramer (Ed.), *Auditory display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display.*, Santa Fé Institute, Santa Fé: Addison-Wesley, pp. xv - xxi.

30. Bregman, A.S. (1990). *Auditory Scene Analysis*. Cambridge, Massachusetts: MIT Press.

31. Brewster, S.A. (1992). *Providing a model for the use of sound in user interfaces* (Technical Report No. YCS 169). University of York, Department of Computer Science.

32. Brewster, S.A., Wright, P.C. & Edwards, A.D.N. (1992). A detailed investigation into the effectiveness of earcons. In G. Kramer (Ed.), *Auditory display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display*, Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 471-498.

33. Brewster, S.A., Wright, P.C. & Edwards, A.D.N. (1993). An evaluation of earcons for use in auditory human-computer interfaces. In S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel, & T. White (Ed.), *INTERCHI'93,* Amsterdam: ACM Press, Addison-Wesley, pp. 222-227.

34. Brewster, S.A., Wright, P.C. & Edwards, A.D.N. (1993). Parallel earcons: Reducing the length of audio messages. *Submitted to the International Journal of Man-Machine Studies*.

35. Brewster, S.A., Wright, P.C. & Edwards, A.D.N. (1994). The design and evaluation of an auditory-enhanced scrollbar. In B. Adelson, S. Dumais, & J. Olson (Ed.), *Proceedings of CHI'94*, Boston, Massachusetts: ACM Press, Addison-Wesley, pp. 173-179.

36. Brown, M.L., Newsome, S.L. & Glinert, E.P. (1989). An experiment into the use of auditory cues to reduce visual workload. In *Proceedings of CHI'89*, Austin, Texas: ACM Press, Addison-Wesley, pp. 339-346.

37. Burgess, D. (1992). *Techniques for low cost spatial audio* (Technical Report No. GIT-GVU-92-09). Graphics, Visualization & Usability Centre, Georgia Institute of Technology.

38. Buxton, W. (1989). Introduction to this special issue on nonspeech audio. *Human Computer Interaction*, 4(1), pp. 1-9.

39. Buxton, W., Gaver, W. & Bly, S. (1991). Tutorial number 8: The use of non-speech audio at the interface. In *Proceedings of CHI'91*, New Orleans: ACM Press: Addison-Wesley.

40. Byers, J.C., Bittner, A.C. & Hill, S.G. (1989). Traditional and raw task load index (TLX) correlations: Are paired comparisions necessary? In A. Mital (Ed.), *Advances in industrial ergonomics*, pp. 481-485. Taylor & Francis.

41. Chowning, J. (1975). Synthesis of complex audio spectra by means of frequency modulation. *Journal of the Audio Engineering Society,* 21(7), pp. 526-534.

42. Cohen, J. (1992). Monitoring background activities. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display.*, Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 499-532.

43. Cohen, J. (1993). "Kirk Here": Using genre sounds to monitor background activity. In S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel, & T. White (Eds.), *INTERCHI'93*, Adjunct Proceedings Amsterdam: ACM Press, Addison Wesley, pp. 63-64.

44. Cohen, M. (1993). Throwing, pitching and catching sound: Audio windowing models and modes. *International Journal of Man-Machine Studies*, 39, pp. 269-304.

45. Cohen, M. & Ludwig, L.F. (1991). Multidimensional audio window management. *International Journal of Man-Machine Studies,* 34, pp. 319-336.

46. Colquhoun, W.P. (1975). Evaluation of auditory, visual and dual-mode displays for prolonged sonar monitoring tasks. *Human Factors*, 17, pp. 425-437.

47. Corcoran, D., Carpenter, A., Webster, J. & Woodhead, M. (1968). Comparison of training techniques for complex sound identification. *Journal of the Acoustical Society of America*, 44, pp. 157-167.

48. Deatherage, B.H. (1972). Auditory and other forms of information presentation. In H. P. Van Cott & R. G. Kinkade (Eds.), *Human engineering guide to equipment design*, pp. 123-160. Washington D.C.: U.S. Government printing office.

49. Deutsch, D. (1980). The processing of structured and unstructured tonal sequences. *Perception and Psychophysics*, 28(5), pp. 381-389.

50. Deutsch, D. (1982). *Psychology of music*. London: Academic Press.

51. Deutsch, D. (1983). Auditory illusions, handedness and the spatial environment. *Journal of the Audio Engineering Society*, 31(9), pp. 607-620.

52. Deutsch, D. (1986). Auditory pattern recognition. In K. R. Boff, L. Kaufman, & P. Thomas (Eds.), *Handbook of perception and human performance*, pp. 32.1-32.49. New York: Wiley.

53. Dewar, K.M., Cuddy, L.L. & Mewhort, D.J. (1977). Recognition of single tones with and without context. *Journal of Experimental Psychology: Human Learning and Memory*, 3(1), pp. 60-67.

54. DiGiano, C.J. (1992) *Visualizing Program Behavior Using Non-speech Audio*. MSc. Thesis, Department of Computer Science, University of Toronto.

55. DiGiano, C.J. & Baecker, R.M. (1992). Program Auralization: Sound Enhancements to the Programming Environment. In *Proceedings of Graphics Interface'92*, pp. 44-52.

56. DiGiano, C.J., Baecker, R.M. & Owen, R.N. (1993). LogoMedia: A sound-enhanced programming environment for monitoring program behaviour. In S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel, & T. White (Ed.), *INTERCHI'93*, Amsterdam: ACM Press, Addison-Wesley, pp. 301-302.

57. Dix, A., Finlay, J., Abowd, G. & Beale, R. (1993). Chapter 9.4 Status/Event Analysis. In *Human-Computer Interaction*, pp. 325-334. London: Prentice-Hall.

58. Dix, A., Finlay, J., Abowd, G. & Beale, R. (1993). *Human-Computer Interaction*. London: Prentice-Hall.

59. Dix, A.J. (1991). Chapter 10: Events and Status. In *Formal Methods for Interactive Systems*, pp. 239-270. London: Academic Press.

60. Dix, A.J. (1992). Beyond the Interface. In *Proceedings of IFIP TC2/WG2.7 Working Conference on Engineering for Human-Computer Interaction,10-14 August 1992*, Ellivuori, Finland.

61. Dix, A.J. & Brewster, S.A. (1994). Causing trouble with buttons. In *Ancilliary Proceedings of HCI'94*, Sterling, UK: Cambridge University Press.

62. Edwards, A.D.N. (1987) *Adapting user interfaces for visually disabled users*. PhD Thesis, Open University, Milton Keynes.

63. Edwards, A.D.N. (1989). Soundtrack: An auditory interface for blind users. *Human Computer Interaction*, 4(1), pp. 45-66.

64. Edworthy, J., Loxley, S. & Dennis, I. (1991). Improving auditory warning design: Relationships between warning sound parameters and perceived urgency. *Human Factors*, 33(2), pp. 205-231.

65. Edworthy, J., Loxley, S., Geelhoed, E. & Dennis, I. (1989). The perceived urgency of auditory warnings. *Proceedings of the Institute of Acoustics*, 11(5), pp. 73-80.

66. European Telecommunications Standards Institute (1992). *Guidelines for the specification of tones* (Technical Report No. HF Temp. Doc. 10, Annex 6.). European Telecommunications Standards Institute.

67. European Telecommunications Standards Institute (1992). *Human Factors (HF); Specification of characteristics of telephone services tones when locally generated in terminals* (Draft ETS 300 295 No. DE/HF-1003-B). European Telecommunications Standards Institute.

68. Fitch, W.T. & Kramer, G. (1992). Sonifying the body electric: Superiority of an auditory over a visual display in a complex, multivariate system. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display*, Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 307-326.

69. Foster, S.H., Wenzel, E.M. & Taylor, R.M. (1991). Real-time synthesis of complex acoustic environments. In *IEEE workshop on applications of signal processing to audio & acoustics, Oct. 20-23,* New Paltz, N.Y.

70. Fraisse, P. (1981). Multisensory aspects of rhythm. In R. D. Walk & H. L. Pick (Eds.), *Intersensory Perception and Sensory-Integration*, pp. 217-245. New York: Plenum Press.

71. Fraisse, P. (1982). Rhythm and tempo. In D. Deutsch (Ed.), *The psychology of music*, pp. 149-180. San Diego, CA.: Academic Press.

72. Frysinger, S.P. (1990). Applied research in auditory data representation. In D. Farrell (Ed.), *Extracting meaning from complex data: processing, display, interaction. Proceedings of the SPIE/SPSE symposium on electronic imaging.* 1259 Springfield, VA.: SPIE, pp. 130-139.

73. Gaver, W. (1986). Auditory Icons: Using sound in computer interfaces. *Human Computer Interaction*, 2(2), pp. 167-177.

74. Gaver, W. (1989). The SonicFinder: An interface that uses auditory icons. *Human Computer Interaction*, 4(1), pp. 67-94.

75. Gaver, W. (1992). Using and creating auditory icons. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display,* Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 417-446.

76. Gaver, W. (1993). Synthesizing auditory icons. In S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel, & T. White (Ed.), *INTERCHI'93,* Amsterdam: ACM Press, Addison-Wesley, pp. 228-235.

77. Gaver, W., Moran, T., MacLean, A., Lövstrand, L., Dourish, P., Carter, K. & Buxton, W. (1992). Realizing a video environment: EuroParc's RAVE system. In P. Bauersfeld, J. Bennett, & G. Lynch (Eds.), *Proceedings of CHI'92,* Monterey, California: ACM Press, Addison-Wesley, pp. 27-35.

78. Gaver, W. & Smith, R. (1990). Auditory icons in large-scale collaborative environments. In D. Diaper, D. Gilmore, G. Cockton, & B. Shackel (Eds.), *Human Computer Interaction: Interact'90,* Cambridge, UK: Elsevier Science Publishers B.V. (North Holland), pp. 735-740.

79. Gaver, W., Smith, R. & O'Shea, T. (1991). Effective sounds in complex systems: The ARKola simulation. In S. Robertson, G. Olson, & J. Olson (Eds.), *Proceedings of CHI'91,* New Orleans: ACM Press, Addison-Wesley, pp. 85-90.

80. Gelfand, S.A. (1981). *Hearing: An introduction to psychological and physiological acoustics*. New York: Marcel Dekker Inc.

81. Gerhing, B. & Morgan, D. (1990). Applications of Binaural Sound in the Cockpit. *Speech Technology,* 5(2), pp. 46-50.

82. Gerth, J.M. (1992) *Performance based refinement of a synthetic auditory ambience: identifying and discriminating auditory sources*. PhD. Thesis, Georgia Institute of Technology.

83. Gleitman, H. (1981). *Psychology*. New York: W. W. Norton & Co.

84. Glinert, E. & Blattner, M. (1992). Programming the multimodal interface. In *ACM MultiMedia'93*: ACM Press, Addison-Wesley, pp. 189-197.

85. Gravetter, F.J. & Wallnau, L.B. (1985). *Statistics for the behavioural sciences* (2nd ed.). St Paul, MN.: West Publishing Company.

86. Grey, J.M. (1977). Multidimensional perceptual scaling of musical timbres. *Journal of the Acoustical Society of America*, 61(5), pp. 1270-1277.

87. Grey, J.M. & Gordon, J.W. (1978). Perceptual effects of spectral modifications on musical timbres. *Journal of the Acoustical Society of America*, 63(5), pp. 1493-1500.

88. Handel, S. (1989). *Listening: An introduction to the perception of auditory events*. Cambridge, Massachusetts: MIT Press.

89. Hapeshi, K. & Jones, D. (1992). Interactive multimedia for instruction: A cognitive analysis of the role of audition and vision. *International Journal of Human-Computer Interaction*, 4(1), pp. 79-99.

90. Harrison, M. & Barnard, P. (1993). On defining requirements for interaction. In *Proceedings of the IEEE International Workshop on requirements engineering*, pp. 50-54. New York: IEEE.

91. Hart, S. & Staveland, L. (1988). Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In P. Hancock & N. Meshkati (Eds.), *Human mental workload*, pp. 139-183. Amsterdam: North Holland B.V.

92. Hart, S.G. & Wickens, C. (1990). Workload assessment and prediction. In H. R. Booher (Ed.), *MANPRINT, an approach to systems integration*, pp. 257-296. New York: Van Nostrand Reinhold.

93. Hartson, H. & Gray, P. (1992). Temporal aspects of tasks in the User Action Notation. *Human-Computer Interaction*, 7, pp. 1-45.

94. Hartson, H.R., Siochi, A.C. & Hix, D. (1990). The UAN: A user-oriented representation for direct manipulation interface designs. *ACM Transactions on Information Systems,* 8(3), pp. 181-203.

95. Hoare, C.A.R. (1985). *Communicating sequential processes*. Exeter, UK: Prentice-Hall International.

96. Iverson, W. (1992). The sound of science. *Computer Graphics World*, 15(1), pp. 54-62.

97. Johnson, J. (1990). Modes in Non-Computer Devices. *International Journal of Man-Machine Studies*, 32(4), pp. 423-438.

98. Johnson, J. & Engelbeck, G. (1989). Modes Survey Results. *ACM SIGCHI Bulletin*, 20(4), pp. 38-50.

99. Jones, D. (1989). The Sonic Interface. In M. Smith & G. Salvendy (Eds.), *Work with computers: Organizational, Management, Stress and health aspects,*. Amsterdam: Elsevier Science publishers.

100. Jones, S.D. & Furner, S.M. (1989). The construction of audio icons and information cues for human-computer dialogues. In T. Megaw (Ed.), *Contemporary Ergonomics: Proceedings of the Ergonomics Society's 1989 Annual Conference,* Reading, UK: Taylor & Francis, pp. 436-441.

101. Kishi, N. (1992). SimUI: Graphical user interface evaluation using playback. In *Proceedings of the Sixteenth Annual International Computer Software & Applications Conference,* Chicago, Illinois: IEEE Computer Society, pp. 121-127.

102. Kramer, G. (Ed.). (1992). *Auditory display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display*. Reading, Massachusetts: Santa Fé Institute, Addison-Wesley.

103. Kramer, G. (1992). An introduction to auditory display. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display,* Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 1-77.

104. Kramer, G. (1992). Some organizing principles for representing data with sound. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display,* Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 185-222.

105. Lee, W.O. (1992). The effects of skill development and feedback on action slips. In A. Monk, D. Diaper, & M. D. Harrison (Eds.), *Proceedings of HCI'92*, VII York, UK: Cambridge University Press, pp. 73-86.

106. Leiser, R., Avons, S. & Carr, D. (1989). Paralanguage and human-computer interaction. Part 2: Comprehension of synthesised vocal segregates. *Behaviour and Information Technology*, 8(1), pp. 23-32.

107. Lerdahl, F. (1987). Timbral hierarchies. *Contemporary Music Review*, 2(1), pp 135-160.

108. Levitt, H. & Voroba, B. (1974). Localization. In S. E. Gerber (Ed.), *Introductory Hearing Science: physical and physiological concepts*, pp. 188-196. Philadelphia: W.B. Saunders Company.

109. Loveless, N.E., Brebner, J. & Hamilton, P. (1970). Bisensory presentation of information. *Psychological Bulletin*, 73(3), pp. 161-199.

110. Loy, G. (1985). Musicians make a standard: The MIDI phenomenon. *Computer Music Journal*, 9(4), pp. 8-26.

111. Lucas, P. (1994). An evaluation of the communicative ability of auditory icons and earcons. In G. Kramer (Ed.), *Accepted for publication in the proceedings of ICAD'94,* Santa Fé Institute, Santa Fé, NM.: Addison-Wesley.

112. Ludwig, L.F., Pincever, N. & Cohen, M. (1990). Extending the notion of a window system to audio. *IEEE Computer*, August, pp. 66-72.

113. Mansur, D.L., Blattner, M. & Joy, K. (1985). Sound-Graphs: A numerical data analysis method for the blind. *Journal of Medical Systems*, 9, pp. 163-174.

114. Matoba, H., Hirabayashi, F. & Kasahara, Y. (1989). Issues in auditory interfaces management: An extra channel for computer applications. In M. Smith & G. Salvendy (Eds.), *Work with Computers: Organizational, Management, Stress and health aspects,*. Amsterdam: Elsevier Science publishers.

115. Mayes, T. (1992). The ‘M’ word: Multimedia interfaces and their role in interactive learning systems. In A. D. N. Edwards & S. Holland (Eds.), *Multimedia Interface Design in Education*, pp. 1-22. Berlin: Springer-Verlag.

116. McCormick, E.J. & Sanders, M.S. (1982). *Human factors in engineering and design* (5th ed.). McGraw-Hill.

117. Monk, A. (1986). Mode Errors: A user-centered analysis and some preventative measures using keying-contingent sound. *International Journal of Man-Machine Studies*, 24, pp. 313-327.

118. Moore, B.C. (1989). *An Introduction to the Psychology of Hearing* (2nd ed.). London: Academic Press.

119. Mountford, S.J. & Gaver, W. (1990). Talking and listening to computers. In B. Laurel (Ed.), *The art of human-computer interface design*, pp. 319-334. Reading, Massachusetts: Addison-Wesley.

120. Myers, B. (1990). All the widgets. *ACM SIGRAPH Video Review*, CHI’90 Special Issue(57).

121. Mynatt, E.D. (1992). Auditory presentation of graphical user interfaces. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display*, Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 533-555.

122. NASA Human Performance Research Group (1987). *Task Load Index (NASA-TLX) v1.0 computerised version*. NASA Ames Research Centre.

123. Norman, D.A. (1986). Chapter 3: Cognitive Engineering. In D. A. Norman & S. W. Draper (Eds.), *User-centered system design*, pp. 31-61. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

124. Norman, D.A. (1988). *The psychology of everyday things*. USA: Basic Books.

125. O’Leary, A. & Rhodes, G. (1984). Cross-modal effects on visual and auditory object perception. *Perception and Psychophysics*, 35(6), pp. 565-569.

126. Open University (1990). *A guide to usability. Part of the Usability Now! Research Technology initiative*. Milton Keynes: DTI and the Open University.

127. Oppenheim, D.V., Anderson, T. & Kirk, R. (1993). Perceptual parameters: Their specification, scoring and control within two software composition systems. In *Proceedings of the International Computer Music Conference, 1993*, Tokyo.

128. Patterson, R.D. (1982). *Guidelines for auditory warning systems on civil aircraft* (CAA Paper No. 82017). Civil Aviation Authority, London.

129. Patterson, R.D. (1989). Guidelines for the design of auditory warning sounds. *Proceeding of the Institute of Acoustics, Spring Conference*, 11(5), pp. 17-24.

130. Patterson, R.D. (1990). Auditory warning sounds in the work environment. In D. E. Broadbent, A. Baddeley, & J. T. Reason (Eds.), *Human Factors in Hazardous Situations. Phil. Trans. B 327*, pp. 485-492. London: The Royal Society.

131. Patterson, R.D., Edworthy, J., Shailer, M., Lower, M. & Wheeler, P. (1986). *Alarm sounds for medical equipment in intensive care areas and operating theatres* (Report No. AC598). Institute of sound and vibration research, University of Southampton.

132. Perrott, D., Sadralobadi, T., Saberi, K. & Strybel, T. (1991). Aurally aided visual search in the central visual field: Effects of visual load and visual enhancement of the target. *Human Factors*, 33(4), pp. 389-400.

133. Pezdeck, K. (1987). Television comprehension as an example of applied research in cognitive psychology. In D. Berger, K. Pezdeck, & W. Banks (Eds.), *Applications in Cognitive Psychology*, pp. 3-15. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

134. Pitt, I. & Edwards, A. (1991). Navigating the interface by sound for blind users. In D. Diaper & N. Hammond (Eds.), *Proceedings of HCI'91*, VI Edinburgh: Cambridge University Press, pp. 373-383.

135. Plomp, R. (1976). Chapter 6: Timbre of complex tones. In *Aspects of tone sensation*, pp. 85-110. London: Academic Press.

136. Portigal, S. (1994) *Auralization of document structure*. MSc. Thesis, The University of Guelph, Canada.

137. Prior, M. & Troup, G.A. (1988). Processing of timbre and rhythm in musicians and non-musicians. *Cortex*, 24(3), pp. 451-456.

138. Rasch, R.A. & Plomp, R. (1982). The perception of musical tones. In D. Deutsch (Ed.), *The Psychology of Music*, pp. 1-21. New York: Academic Press.

139. Rayner, K. & Pollatsek, A. (1989). *The Psychology of Reading*. Englewood Cliffs, New Jersey: Prentice-Hall International, Inc.

140. Reason, J. (1990). *Human Error*. Cambridge, UK: Cambridge University Press.

141. Reber, A.S. (1985). *The Penguin Dictionary of Psychology*. London: Penguin Books.

142. Reich, S.S. (1980). Significance of pauses for speech perception. *Journal of Psycholinguistic Research*, 9(4), pp. 379-389.

143. Reichman, R. (1986). Chapter 14: Communications paradigms for a window system. In D. A. Norman & S. W. Draper (Eds.), *User-Centered System Design*, pp. 285-314. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

144. Robson, C. (1994). *Experiment, design and statistics in psychology* (3rd ed.). London: Penguin Books Ltd.

145. Rosenberg, K. (1990). *Statistics for behavioural sciences*. Dubuque, IA.: Wm. C. Brown Publishers.

146. Sakamoto, N., Gotoh, T. & Kimaura, Y. (1976). On ‘out of head localization’ in headphone listening. *Journal of the Acoustic Engineering Society,* 24(9), pp. 710-716.

147. Scharf, B. & Houtsma, A.J. (1986). Audition II: Loudness, pitch, localization, aural distortion and pathology. In K. R. Boff, L. Kaufman, & P. Thomas (Eds.), *Handbook of perception and human performance*, pp. 15.1-15.60. New York: Wiley.

148. Scholes, P.A. (1975). *The oxford companion to music* (10th ed.). Oxford: Oxford University Press.

149. Scott, D. (1993). Status conspicuity, peripheral vision and text editing. *Behaviour and Information Technology,* 12(1), pp. 23-31.

150. Scott, D. & Findlay, J.M. (1991). Optimum display arrangements for presenting status information. *International Journal of Man-Machine Studies,* 35, pp. 399-407.

151. Sellen, A., Kurtenbach, G. & Buxton, W. (1992). The prevention of mode errors through sensory feedback. *Human Computer Interaction*, 7, pp. 141-164.

152. Sellen, A.J., Kurtenbach, G.P. & Buxton, W. (1990). The role of visual and kinesthetic feedback in the prevention of mode errors. In D. Diaper, D. Gilmore, G. Cockton, & B. Shackel (Eds.), *Human Computer Interaction: Interact’90*, Cambridge, UK: Elsevier Science Publishers B.V. (North Holland), pp. 667-673.

153. Slowiaczek, L.M. & Nusbaum, H.C. (1985). Effects of speech rate and pitch contour on the perception of synthetic speech. *Human Factors,* 27(6), pp. 701-712.

154. Smith, B. (1991). UNIX goes Indigo. *Byte*, 16(9), pp. 40-41.

155. Smith, S., Bergeron, R.D. & Grinstein, G.G. (1990). Stereophonic and surface sound generation for exploratory data analysis. In *CHI ‘90,* Seattle, Washington: ACM Press: Addison-Wesley, pp. 125-132.

156. Smither, J. (1993). Short term memory demands in processing synthetic speech by old and young adults. *Behaviour and Information Technology*, 12(6), pp. 330-335.

157. Sonnenwald, D.H., Gopinath, B., Haberman, G.O., Keese, W.M. & Myers, J.S. (1990). InfoSound: An audio aid to program comprehension. *Proceedings of the 23rd Hawaii International Conference on System Sciences*, pp. 541-546.

158. Speeth, S.D. (1961). Seismometer Sounds. *Journal of the Acoustical Society of America*, 33(7), pp. 909-916.

159. Spivey, J.M. (1992). *The Z notation: A reference manual* (2nd ed.). Hemel Hempstead, UK: Prentice Hall International.

160. Stevens, R.D., Brewster, S.A., Wright, P.C. & Edwards, A.D.N. (1994). Providing an audio glance at algebra for blind readers. In G. Kramer (Ed.), *Accepted for publication in the proceedings of ICAD’94,* Santa Fé Institute, Santa Fé: Addison-Wesley.

161. Strybel, T., Manligas, C. & Perrott, D. (1992). Minimum audible movement angle as a function of the azimuth and elevation of the source. *Human Factors*, 34(3), pp. 267-275.

162. Sumikawa, D., Blattner, M. & Greenberg, R. (1986). Earcons: Structured Audio Messages. *Unpublished paper*.

163. Sumikawa, D., Blattner, M., Joy, K. & Greenberg, R. (1986). *Guidelines for the syntactic design of audio cues in computer interfaces* (Technical Report No. UCRL 92925). Lawrence Livermore National Laboratory.

164. Sumikawa, D.A. (1985). *Guidelines for the integration of audio cues into computer user interfaces* (Technical Report No. UCRL 53656). Lawrence Livermore National Laboratory.

165. Svean, J. (1994). AEP: Active Ear Plug. Sintef Research Laboratory, Norway. *Personal Communication.*

166. Swift, C.G., Flindell, I.H. & Rice, C.G. (1989). Annoyance and impulsivity judgements of environmental noises. *Proceedings of the Institute of Acoustics*, 11(5), pp. 551-559.

167. Tessler, L. (1981). The SmallTalk environment. *Byte*(August), pp. 90-147.

168. Thimbleby, H. (1990). *User Interface Design*. New York: ACM Press, Addison-Wesley.

169. Vanderveer, N.J. (1979). *Ecological acoustics: Human perception of environmental sounds* (Thesis No. 40/09B, 4543). Dissertation Abstracts International.

170. Von Bismarck, G. (1974). Timbre of steady sounds: A factorial investigation of its verbal attributes. *Acustica*, 30(3), pp. 146-159.

171. Wagenaar, W.A., Varey, C.A. & Hudson, P.T. (1984). Do audiovisuals aid? A study of bisensory presentation on the recall of information. In H. Bouma & D. G. Bouwhuis (Eds.), *Attention and Performance: X*, pp. 379-391. Lawrence Erlbaum Associates.

172. Walker, J.T. & Scott, K.J. (1981). Auditory-visual conflicts in the perceived duration of lights, tones and gaps. *Journal of Experimental Psychology: Human Perception and Performance*, 7(6), pp. 1327-1339.

173. Warren, W.H. & Verbrugge, R.R. (1984). Auditory perception of breaking and bouncing events: A case study in ecological acoustics. *Journal of Experimental Psychology: Human Perception and Performance*, 10, pp. 704-712.

174. Watkins, W.H. & Feehrer, C.E. (1965). Acoustic facilitation of visual detection. *Journal of Experimental Psychology*, 70(3), pp. 322-333.

175. Webster, B. (1989). *The NeXT Book*. Reading, MA: Addison-Wesley.

176. Wedin, L. & Goude, G. (1972). Dimension analysis of the perception of instrumental timbre. *Scandinavian Journal of Psychology*, 13(3), pp. 228-240.

177. Wenzel, E., Wightman, F. & Foster, S. (1988). Development of a 3D auditory display system. *SIGCHI Bulletin*, 20(2), pp. 52-57.

178. Wenzel, E., Wightman, F.L. & Kistler, D. (1991). Localization with non-individualized virtual display cues. In S. Robertson, G. Olson, & J. Olson (Eds.), *CHI'91*, New Orleans: ACM Press, Addison-Wesley, pp. 351-359.

179. Wenzel, E.M. (1992). Three-Dimensional virtual acoustic displays. *Presence: teleoperators and virtual environments*, 1, pp. 80-107.

180. Wenzel, E.M., Foster, S.H., Wightman, F.L. & Kistler, D.J. (1989). Realtime digital synthesis of localized auditory cues over headphones. In *IEEE workshop on applications of signal processing to audio & acoustics. Oct. 15-18,* New Paltz, N.Y.

181. Wessell, D.L. (1979). Timbre space as a musical control structure. *Computer Music Journal*, 3(2), pp. 42-52.

182. Wickens, C.D., Mountford, S.J. & Schreiner, W. (1981). Multiple resources, task-hemispheric integrity and individual differences in time-sharing. *Human Factors*, 23(2), pp. 211-229.

183. Williams, S. (1992). Perceptual principles in sound grouping. In G. Kramer (Ed.), *Auditory Display, sonification, audification and auditory interfaces. The Proceedings of the First International Conference on Auditory Display,* Santa Fé Institute, Santa Fé: Addison-Wesley, pp. 95-126.

184. Wright, P.C. & Monk, A.F. (1989). Evaluation for design. In A. Sutcliffe & L. Macaulay (Eds.), *People and computers 5*, pp. 345-358. Cambridge: Cambridge University Press.

185. Yager, T. (1991). The Littlest SPARC. *Byte*, 16(2), pp. 169-174.