Chapter 12. Non-Speech Auditory Output

1 Introduction and a brief history of non-speech sound and HCI

The combination of visual and auditory information at the human-computer interface is powerful tool for interaction. In everyday life both senses combine to give complementary information about the world. Our visual system gives us detailed information about a small area of focus whereas our auditory system provides general information 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. Blattner & Dannenberg (1992) 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 organizations 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 verbalized 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 by the addition of non-speech auditory output to standard graphical displays (see Chapter 14 for more on multimodal interaction). 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 using sound. 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 capitalize on the interdependence between them and present information in the most efficient way possible.

The classical uses of non-speech sound can be found in the human factors literature (see McCormick & Sanders, 1982). Here it is used mainly for alarms and warnings or monitoring and status information. Buxton (1989) extends these ideas and suggests that encoded messages could be used to present more complex information in sound and it is this type of auditory feedback that will be considered here.

The other main use of non-speech sound is in music and sound effects for games and other multimedia applications. These kinds of sounds indicate to the user something about what is going on and try to create a mood for the piece (much as music does in film and radio). Work on auditory output for interaction takes this further and uses sound to present information: things that the user might not otherwise see or important events that the user might not notice.

The use of sound to convey information in computers is not new. In the early days of computing programmers used to attach speakers to their computer's bus or program counter (Thimbleby, 1990). The speaker would click each time the program counter was changed. Programmers would get to know the patterns and rhythms of sound and could recognize 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 may not see.

Two important events kick-started the research area of non-speech auditory output: the first was the special issue of the HCI journal in 1989 on non-speech sound, edited by Bill Buxton (Buxton, 1989). This laid the foundations for some of the key work in the area; it included papers by Blattner on *Earcons* (Blattner, Sumikawa, & Greenberg, 1989), Gaver on *Auditory Icons* (Gaver, 1989) and Edwards on *Soundtrack* (Edwards, 1989). The second event was the First International

Conference on Auditory Display (ICAD'92) held in Santa Fe in 1992 (Kramer, 1994a). For the first time this meeting brought together the main researchers interested in the area to get discussion going and the proceedings of this meeting (Kramer, 1994a) are still a valuable resource today (see the ICAD Website at www.icad.org for the proceedings of the ICAD conferences). Resulting from these two events has been a large growth in research in the area during the 1990's.

One active theme of research is *sonification* – literally visualization in sound (Kramer & Walker, 1999). A review of this area is beyond the scope of this chapter but the interested reader is referred to the ICAD Website for more details on this topic.

The rest of this chapter goes into detail in all aspects of auditory interface design. Section 2 presents some of the advantages and disadvantages of using sound at the interface. Then follows a brief introduction to psychoacoustics, or the study of the perception of sound. Section 4 gives information about the basic sampling and synthesis techniques needed for auditory interfaces. Section 5 describes the main techniques that are used for auditory information presentation and Section 6 then goes through some of the main applications of sound in human-computer interaction:

- Sonic enhancement and graphical replacement in GUIs
- Sound for users with visual impairments
- Sound for wearable and mobile computers

The chapter finishes with some conclusions about the state of research in this area.

2 Why use non-speech sound in human-computer interfaces?

2.1 Some advantages offered by sound

It is important to consider why non-speech sound should be used as an interaction technique. There are many reasons why it is advantageous to use sound in user interfaces:

Vision and hearing are interdependent. Our visual and auditory systems work well together. Our eyes provide high-resolution information around a small area of focus (with peripheral vision extending further). According to Perrott, Sadralobadi, Saberi & Strybel (1991) 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 (Rayner & Pollatsek, 1989). Sounds, on the other hand, can be heard from all around the user: above, below, in front or behind, but with a much lower resolution. Therefore 'our ears tell our eyes where to look': if there is an interesting sound from outside our view we will turn to look at it to get more detailed information.

Superior temporal resolution. As Kramer (1994b) says "Acute temporal resolution is one of the greatest strengths of the auditory system". In certain cases, reactions to auditory stimuli have been shown to be faster than reactions to visual stimuli (Bly, 1982).

Reduce the load on the user's visual system. Modern, large or multiple screen graphical interfaces use the human visual system very intensively. This means that we may miss important information because our visual system is overloaded – we have just too much to look at. To stop this overload, information could be displayed in sound so that the load could be shared between senses.

Reduce the amount of information needed on screen. Related to the point above is the problem with information presentation on devices with small visual displays, such as mobile telephones, or personal digital assistants (PDAs). These have very small screens that can easily become cluttered. To solve this some information could be presented in sound to release screen space.

Reduce demands on visual attention. Another issue with mobile devices is that users who are using them on the move cannot devote all of their visual attention to the device – they must look where they are going to avoid traffic, pedestrians, etc. In this case visual information may be missed because the user is not looking at the device. If this was played in sound then the information would be delivered wherever the user was looking.

The auditory sense is under-utilized. The auditory system is very powerful and would appear to be able to take on extra capacity (Bly, 1982). We can listen to (and some can compose) highly complex musical structures. As Alty (1995) says "The information contained in a large musical work (say a symphony) is very large… The information is highly organized into complex structures and sub-structures. The potential therefore exists for using music to successfully transmit complex information to a user".

Sound is attention grabbing. Users can choose not to look at something but it is harder to avoid hearing it. This makes sound very useful for delivering important information.

Some objects or actions within an interface may have a more natural representation in sound. Bly (1982) 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 understand information in different ways.

To make computers more usable by visually disabled users. With the development of graphical displays user interfaces became much harder for visually impaired users to operate. A screen reader (see Section 6.2 below) cannot easily read this kind of graphical information. Providing information in an auditory form can help solve this problem and allow visually disabled persons to use the facilities available on modern computers.

2.2 Some problems with non-speech sound

The above reasons show that there are advantages to be gained from adding sound to human-computer interfaces. However, Kramer (1994b) suggests some general difficulties with sound:

Low resolution: Many auditory parameters are not suitable for high-resolution display of quantitative information. Using volume, for example, only a very few different values can be unambiguously presented (Buxton, Gaver, & Bly, 1991). 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 more than 30 degrees to the side (see Section 4.4 and Begault, 1994). 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 he/she can tell if a value is going up or down. It is very difficult to present absolute data unless the listener has perfect pitch (which is rare). 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. For example, changing the pitch of a note may affect its perceived loudness and *vice versa* (see Section 3).

Transience of information: Sound disappears when it has been presented and can therefore cause problems. Users must remember the information that the sound contained or some method of re-playing must be provided. In vision the user can easily look back at the display and see the information again (this is not always the case - think, for example, of an air conditioning system: its sounds continue for long periods of time and become habituated. Sounds often continue in the background and only become apparent when they change in some way).

Annoyance due to auditory feedback: There is one problem with sound that has not yet been mentioned but it is the one most commonly brought up against the use of sound in user interfaces: *annoyance*. As this is such an important topic it will be discussed in detail in Section 2.4.

2.3 Comparing speech and non-speech sounds for interface design

One obvious question is: why not use speech for output? Why do we need to use non-speech sounds? Many of the advantages presented above apply to speech as well as non-speech sounds. There are, however, some advantages to non-speech sounds. If we think of a visual analogy speech output is like the text on a visual display and non-speech sounds are like the icons. Presenting information in speech is slow because of its serial nature; to assimilate information the user must typically 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 (although the user might have to learn the meaning of the non-speech sound whereas the meaning is contained within the speech and so requires no learning – just like the visual case). Speech suffers from many of the same problems as text in text-based computer systems, as this is also a serial medium. Barker & Manji (1989) 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 (Barker & Manji, 1989). In the same way, an encoded sound message can communicate its information in fewer sounds. The user hears the sound then recalls its meaning rather than having the meaning described in words. Work has been done on increasing the presentation rate of synthetic speech (Aldrich & Parkin, 1989). This found that the accuracy of recognition decreased as the speech rate went up. Pictorial icons can also be more universal: they can mean the same thing in many different languages and cultures, and the non-speech sounds have similar universality (given some of the different musical cultures).

An important ability of the auditory system is *habituation* where continuous sounds can fade into the ‘background’ of consciousness after a short period of time. If the sound changes (or stops) then it would come to the foreground of attention because of the sensitivity of the auditory system to change (Buxton, 1989). Habituation is difficult to achieve with speech because of the large dynamic range it uses. According to Patterson (1982): “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”. It is easier to habituate certain types of non-speech sounds (think of an air conditioner where you only notice that it was on when it switches off) and sounds can be designed to facilitate habituation if required.

Baddeley (1990) gives evidence to show that background speech, even at low intensities, is much more disruptive than non-speech sound when recalling information. He 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. 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 and, as we saw above, this can cause problems with the ability to hear consonants.

Non-speech sounds are also good for presenting continuous information, for example a graph of stock market data. In speech particular values could be spoken out at particular times but there would be no way to monitor the overall trend of the data. Methods for doing this in non-speech sounds were developed over 15 years ago (Mansur, Blattner, & Joy, 1985) and have proved to be very effective (see Sound graphs in Section 6.2 below for more on this).

This discussion has shown that there are many reasons to think of using non-speech sounds in addition to speech in human-computer interfaces. There are as yet very few interfaces that make good use of both. Speech in general is good for giving instructions and absolute values, non-speech sounds are good at giving rapid feedback on actions, presenting highly structured information quickly and presenting continuous data. Together they make a very effective means of presenting information non-visually.

2.4 Avoiding annoyance

The main concern potential users of auditory interfaces have is annoyance due to sound pollution. There are two aspects to annoyance: A sound may be annoying to the user whose machine is making the noise (the primary user) and/or annoying to others in the same environment who overhear it (secondary users). Buxton (1989) has discussed some of the problems of sound and suggests that some sounds help us (information) and some impede us (noise). We therefore need to design sounds so that there are more informative ones and less noise. Of course, one person's informative sounds are another's noise so it is important to make sure that the sounds on one computer are not annoying for a colleague working nearby.

There are few studies that particularly look at the problems of annoyance due to non-speech sounds in computers. There are, however, many studies of annoyance from speech (e.g. Berglund, Harder, & Preis, 1994), from the sounds of aircraft, traffic or other environmental noise and most of these suggest that the primary reason for the annoyance of sound is excessive volume. In a different context, Patterson (1989) investigated some of the problems with auditory warnings in aircraft cockpits. Many of the warnings were added in a 'better safe than sorry' manner that lead to them being so loud that the pilot's first response was to try and turn them off rather than deal with the problem being indicated. One of Patterson's main recommendations was that the volume of the warnings should be reduced.

A loud sound therefore grabs the attention of the primary user, even when the sound is communicating an unimportant event. As the sound is loud, it travels from one machine to the ears of other people working nearby, increasing the noise in their environment.

So, how can annoyance be avoided? One key way is to avoid using intensity as a cue in sound design for auditory interfaces. Quiet sounds are less annoying. Listeners are also not good at making absolute intensity judgments (see the next section). Therefore, intensity is not a good cue for differentiating sounds anyway.

Headphones could be used so that sounds are only heard by the primary user. This may be fine for users of mobile telephones and music players but is not always a good solution for desktop users who do not want to by physically tied to their desks or cut-off from their colleagues.

Manipulating sound parameters other than intensity can make sounds attention grabbing (but not annoying). Rhythm or pitch can be used to make sounds demanding because the human auditory system is very good at detecting changing stimuli (for more see Edworthy, Loxley, & Dennis, 1991). Therefore if care is taken with the design of sounds in an interface, specifically avoiding the use of volume changes to cue the user, then many of the problems of annoyance due to sound can be avoided and the benefits of non-speech sound can be taken advantage of.

3 Perception of sound

This section will provide some basic information about the perception of sound that is applicable to non-speech auditory output. The auditory interface designer must be conscious of the effects of psychoacoustics, or the perception of sound, when designing sounds for the interface. As Frysinger (1990) says: "The characterization 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. There is not enough space here to give great detail on this complex area, Chapter 26 has more on the basics of human auditory perception and for more detail in general see Moore (1997).

An important first question is: What is sound? Sounds are pressure variations that propagate in an elastic medium (in this case, the air). The pressure variations originate from the motion or vibration of objects. These pressure variations are what hit the listener's ear and start the process of perceiving the sound. The pressure variations plotted against time can be seen in Figure 1. This shows the simplest form of sound: a sine wave (as might be produced by a tuning fork). A sound is made up from three basic components:

Frequency is the number of times per second the wave repeats itself (Figure 1 shows three cycles). It is normally measured in Hertz (Hz). *Amplitude* is the deviation away from the mean pressure level, or force per unit area of a sound. It is normally measured in decibels (dB). *Phase:* The position of the start of the wave on the time axis (measured in milliseconds).

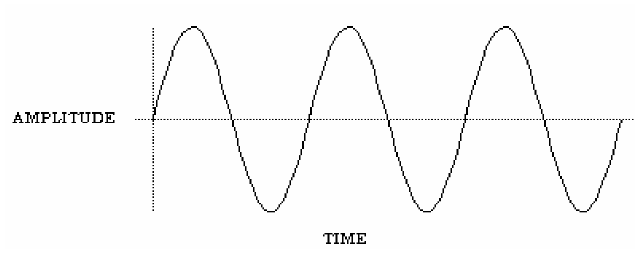

Figure 1: A sine wave.

Sounds from the real world are normally much more complex than Figure 1 and tend to be made up of many sine waves with different frequencies, amplitudes and phases. Figure 2 shows a more complex sound made of three sine wave components (or *partials*) and the resulting waveform (in a similar way as a sound might be created in an additive synthesis system – see Section 4.1 below). *Fourier analysis* allows a sound to be broken down into its component sine waves (Gelfand, 1981).

The sounds in Figures 1 and 2 are *periodic* – they repeat regularly over time. This is very common for many types of musical instruments that might be used in an auditory interface. Many natural, everyday sounds (such as impact sounds) are not periodic and do not keep repeating. The sound in Figure 2 is also *harmonic* – its partials are integer multiples of the lowest (or *fundamental*) frequency. This is again common for musical instruments but not for everyday sounds. Periodic harmonic sounds have a recognizable pitch; where as non-periodic, inharmonic sounds tend to have no clear pitch.

The attributes of sound described above are the physical aspects. There is a corresponding set of perceptual attributes (or more on each of these perceptual attributes see Chapter 26). *Pitch* is the perceived frequency of a sound. Pitch is roughly a logarithmic function of frequency. It can be defined as the attribute of auditory sensation in terms of which sounds may be ordered on a musical scale (Moore, 1997). 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. It is one of the most useful and easily controlled aspects of sound and is very useful for auditory interface designers. However, as Buxton *et al.* (1991) say "It is important to be aware of the myriad interactions between pitch and other attributes of sound …". For example, pitch is affected by loudness: at less than 2kHz an increase in intensity increases the perceived pitch, at 3kHz and over an increase in intensity decreases the perceived pitch

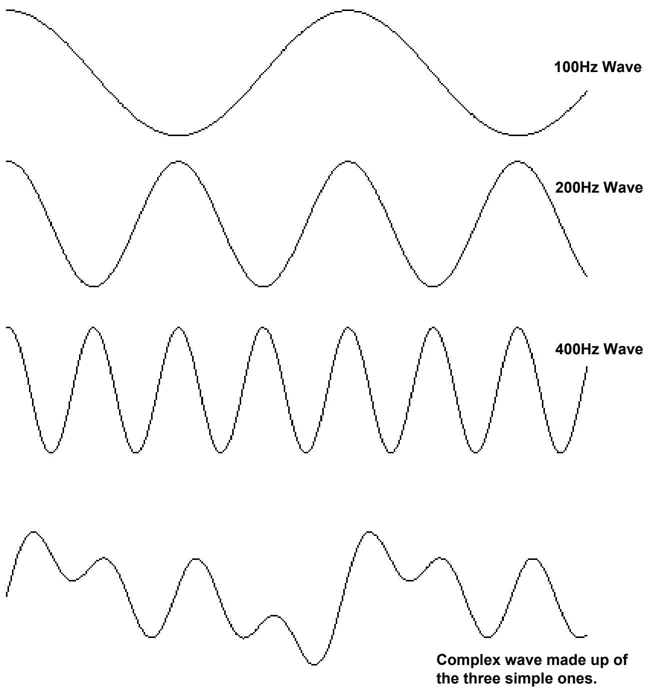

Figure 2: A complex wave made up of three components with its fundamental at 100Hz.

Humans can perceive a wide range of frequencies. The maximum range we can hear is from 20Hz to 20kHz. This decreases with age so that at age 70 a listener might only hear a maximum of 10kHz. It is therefore important to make sure that the sounds in an auditory interface are perceivable by its users (also poor quality loudspeakers may not be able to cope with the highest frequencies). Listeners are not good at making absolute judgments of pitch (Moore, 1997). Only 1% of the population has *perfect pitch*. Another important factor is *tone deafness.* Moore suggests that this is a misnomer and almost everyone is able to tell that two sounds are different; they are not always able to say which is higher or lower. This can often be overcome with practice but it is important for the auditory interface designer to be aware of the problem. Mansur, Blattner & Joy (1985) give evidence of one other important effect: "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 is important 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 then pitch could provide them in the vertical plane. Guidelines for the use of pitch (and the other parameters below) are described in Section 5.2.

Loudness is the perceived intensity of a sound. Loudness (L) is related to intensity (I) according to the *Power Law*: $L = k I^{0.3}$ (Gelfand, 1981). Therefore a 10dB increase in intensity doubles the perceived loudness of a sound. Loudness is again affected by the other parameters of sound. For example, sounds of between 1kHz and 5kHz sound louder at the same intensity level than those outside that frequency range. Humans can perceive a very wide range of intensities: the most intense sound that a listener can hear is 120dB louder than the quietest. This equates to a ratio of 1,000,000,000,000:1 (Moore, 1997). Buxton *et al.* (1991) also report that listeners are "very bad at making absolute judgments about loudness" and also "Our ability to make relative judgments of loudness are limited to a scale of about three different levels". It is also a primary cause of annoyance (see Section 2.4 above) so should be used sparingly by auditory interface designers.

Timbre is the 'quality' of the sound. It is the attribute of auditory sensation in terms of which a listener can judge two sounds with the same loudness and pitch to be dissimilar. It is what makes a violin sound different to a piano even if both are playing the same pitch at the same loudness. Very little is known about its structure or dimensions. It is known to be based partly on the spectrum and dynamics of a sound. Even though its structure is not well understood it is one of the most important attributes of sound that an interface designer can use. As Blattner, Sumikawa and Greenberg (1989) say "Even though timbre is difficult to describe and notate precisely, it is one of the most immediate and easily recognizable characteristics of sound" (both *auditory icons* and *earcons* use timbre as one of their fundamental design attributes – see Section 4.4 below). Many of the synthesis techniques in the next section make it easy for a designer to create and use different timbres.

Duration is another important attribute of sound. Sounds of different durations are used to form rhythmic structures that are a fundamental part of music, for example. Duration can also affect the other parameters of sound. For example, 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 taking place so they must be made loud enough for listeners to hear.

Direction is the position of the sound source. This is often overlooked but is an important aspect of sound in our everyday lives. As mentioned above, one of the key differences between sight and hearing is that sounds can be heard as coming from all around the listener. If a sound source is located to one side of the head then the sound reaching the further ear will be reduced in intensity (*Interaural Intensity Difference* - IID) and delayed in time (*Interaural Time Difference* - ITD) (Begault, 1994). These two factors are key in allowing a listener to *localize* a sound in space.

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 and Perrott (1992) report that in the median plane sound sources only 1° apart can be detected. At 90° azimuth (directly opposite one ear) sources must be around 40° apart. This has important implications for auditory displays. It indicates that higher-resolution sounds can be used when presented in front of the user (see Section 4.4 for more on sound positioning).

4 Technology and sound production

Most modern computers have very sophisticated sound hardware available for the auditory interface designer. This is normally for playing games but is sufficient to do most of the things required by an auditory interface. The aim of this section is to briefly describe some of the main technology that is important for designers to understand when creating interfaces.

There are two main aspects to sound production: the first is sound synthesis and the second is sound sampling. A basic overview will be given focusing on aspects related to audio interfaces. For much more detail on sound synthesis and MIDI see Roads (1996) for more on sampling see Pohlmann (1995).

4.1 A brief introduction to sound synthesis and MIDI

The aim of sound synthesis is to generate a sound from a stored model, often a model of a musical instrument. For auditory interfaces we need a wide and good quality range of sounds that we can generate in real-time as the user interacts with the interface. Synthesizers come in three main forms: soundcards on PCs, external hardware devices and it is now also becoming possible to do good quality synthesis in software.

Most synthesizers are *polyphonic*, i.e. able to play multiple notes at the same time (as opposed to *monophonic*). This is important for auditory interface design as you might well want to play a chord made up of several notes. Most modern synthesizers are *multi-timbral*, i.e. can play multiple different instruments at the same time. This is again important as in many situations a sound composed of two different instruments might be required. The main forms of synthesis will now be briefly reviewed.

Wavetable Synthesis is one of the most common and low cost synthesis techniques. Many of the most popular PC soundcards use it (such as the SoundBlaster™ series from Creative Labs and the Santa Cruz™ soundcards from Turtle Beach). The idea behind Wavetable synthesis is to use existing sound recordings (which are often very difficult to synthesize exactly) as the starting point, and create very convincing simulations of acoustical instruments based on them (Heckroth, 1994; Roads, 1996). A sample (recording) of a particular sound will be stored in the soundcard. It can then be played back to produce a sound. The sample memory in these systems contains a large number of sampled sound segments, and can be thought of as a 'table' of sound waveforms that may be looked up and utilized when needed. Wavetable synthesizers employ a variety of different techniques, such as sample looping, pitch shifting, mathematical interpolation, and polyphonic digital filtering, in order to reduce the amount of memory required to store the sound samples or to get more types of sounds. More sophisticated synthesizers contain more wavetables (perhaps one or more for the initial *attack* part of a sound and then more for the *sustain* part of the sound and then more for the final *decay* and *release* parts). Generally, the more wavetables that are used the better the quality of the synthesis, but this does require more storage. It is also possible to combine multiple separately controlled wavetables to create one new instrument.

Wavetable synthesis is not so good if you want to create new timbres as it lacks some of the flexibility of the other techniques below. Most wavetable synthesizers contain many sounds (often many hundreds) so there may not be a great need to create new ones. For most auditory interfaces the sounds from a good quality wavetable synthesizer will be perfectly acceptable.

FM (frequency modulation) Synthesis techniques generally use one periodic signal (the modulator) to modulate the frequency of another signal (the carrier) (Chowning, 1975). If the modulating signal is in the audible range, then the result will be a significant change in the timbre of the carrier signal. Each FM voice requires a minimum of two signal generators. These generators are commonly referred to as 'operators', and different FM synthesis implementations have varying degrees of control over the operator parameters. Sophisticated FM systems may use 4 or 6 operators per voice, and the operators may have adjustable envelopes that allow adjustment of the attack and decay rates of the signal. FM synthesis is cheap and easy to implement and can be useful for creating expressive new synthesized sounds. However, if the goal is to recreate the sound of an existing instrument, then FM synthesis is not the best choice as it can generally be done more easily and accurately with wavetable-based techniques (Heckroth, 1994).

Additive (and Subtractive) Synthesis is the oldest form of synthesis (Roads, 1996). Basically multiple sine waves are added together to produce a more complex output sound (subtractive synthesis is basically the opposite: a complex sound has frequencies filtered out to create the sound required). Using this method it is theoretically possible to create any sound (as all complex sounds can be decomposed into sets of sine waves by Fourier analysis). However, it can be very difficult to create any particular sound. Computer musicians often use this technique as it is very flexible and easy to create new and unusual sounds but may be less useful for auditory interface designers.

Physical Modeling Synthesis uses mathematical models of the physical acoustic properties of instruments and objects. Equations describe the mechanical and acoustic behavior of an instrument. The better the simulation of the instrument the more realistic the sound produced. Mathematical models can be created for a wide range of instruments and these can be manipulated in very flexible ways. Non-existent instruments can also be modeled and made to produce sounds.

Physical modeling is an extremely good choice for synthesis of many classical instruments, especially those of the woodwind and brass families. Its parameters directly reflect the ones of the real instrument and excellent emulations can be produced. The downside is that it can require large amounts of processing power which, in turn, can limit the polyphony of physical modeling synthesizers (Heckroth, 1994).

4.1.1 The Musical Instrument Digital Interface - MIDI

The *Musical Instrument Digital Interface* (MIDI) allows the real-time control of electronic instruments (such as synthesizers, samplers, drum machines, etc.) and is now very widely used. It specifies a hardware interconnection scheme, a method for data communications and a grammar for encoding musical performance information (Roads, 1996). For auditory interface designers the most important part of MIDI is the performance data, which is a very efficient method of representing sounds. Most soundcards support MIDI with an internal synthesizer and also provide a MIDI interface to connect to external devices. Both Apple MacOS and Microsoft Windows both have good support for MIDI. Most programming languages now come with libraries supporting MIDI commands. For example, version 1.3 of the Java programming language from Sun Microsystems has a built in software synthesizer (as part of the Java Sound package) which is directly accessible from Java code.

MIDI performance information is like a piano-roll: notes are set to turn on or off and play different instruments over time. A MIDI message is an instruction that controls some aspect of the performance of an instrument. A MIDI message is made up of a *status byte* which indicates the type of the message, followed by up to two *data bytes* that give the parameters. For example the *Note On* command takes two parameters: one value giving the pitch of the note required and another the velocity. This makes it a very compact form of presentation.

Performance data can be created dynamically from program code or by a sequencer. In an auditory interface the designer might assign a particular note to a particular interface event – for example, a click on a button. When the user clicks on the button a MIDI Note On event will be fired, when the user releases the button the corresponding Note Off event will be sent. This is a very simple and straightforward way of adding sounds to a user interface. With a sequencer, data can be entered using classical music notation by dragging and dropping notes onto a stave, or by using an external piano-style keyboard. This could then be saved to a file of MIDI for later playback (or could be recorded and played back as a sample – see the below).

4.2 A brief introduction to sampling

In many ways sampling is simpler than synthesis. The aim is to make a digital recording of an analogue sound and then to be able to play it back later with the played back sound matching the original as closely as possible. There are two important aspects: *sample rate* and *sample size*.

4.2.1 Sample rate

This is the number of discrete 'snapshots' of the sound that are taken, often measured per second. The higher the sampling rate, the higher the quality of the sound when it is played back. With a low sampling rate few snapshots of the sound are taken and the recording will not match well to the sound being recorded. The *Sampling Theorem* (Roads, 1996) states that "In order to be able to reconstruct a signal, the sampling frequency must be at least twice the frequency of the signal being sampled". As mentioned above, the limit of human hearing is around 20kHz, therefore a maximum rate of 40kHz is required to be able to record any sound that a human can hear (Roads, 1996). The standard audio CD format uses a sample rate of 44.1kHz, meaning that is can record all of the frequencies that a human can hear. If a lower sampling rate is used then higher frequencies are lost. For example the .au sample format uses a sampling rate of 8kHz, meaning that only frequencies of less than 4kHz can be recorded. For more details on the huge range of sample formats see Bagwell (1998).

Higher sampling rates generate much more data than lower ones so may not always be suitable if storage is limited (for example on a mobile computing device). As an auditory interface designer it is important to think about the frequency range of the sounds needed in an interface and this might allow the sample rate to be reduced. If the highest quality is required then you must be prepared to deal with large audio files.

4.2.2 Sample Size

The larger the sample size the better the quality of the recording, as more information is stored at each snapshot of the sound. Sample size defines the volume (or dynamic) range of the sound. With an 8 bit sample only 256 discrete amplitude (or quantization) levels can be represented. To fit an analogue sound into one of these levels might cause it to be rounded up or down and this can add noise to the recording. CD quality sounds use 16 bit samples giving 65536 different levels, so the effects of quantization are reduced. Many high quality samplers use 24 bit samples to reduce the problems of quantization still further.

The main two bit sizes used in most soundcards are 8 and 16 bits. As with sample rates the main issue is size: 16 bit samples require a lot of storage, especially at high sample rates. Audio CD quality sound generates around 10 Mbytes of data per minute.

4.3 Comparing MIDI synthesis to sampling for auditory interface design

MIDI is very flexible as synthesizers can generate sound in real-time as it is needed. If you do not know in advance all of the sounds you might want in your auditory interface then this can very effective – as the sound is needed it is just played by the synthesizer. A system that is based around samples can only play back samples that have been pre-recorded and stored.

Another advantage of MIDI is that sounds can be changed. Once a sample has been stored it is difficult to change it. For example, it is possible to change the speed and pitch of an audio steam independently with MIDI. If a sample is played back at a different speed its pitch will change, which may cause undesirable effects.

MIDI commands are also very small; each command might only take up 2 or 3 bytes. Generating sounds from code in your auditory interface is straightforward. For instance, files containing high quality stereo sampled audio require about 10 Mbytes of data per minute of sound, while a typical MIDI sequence might consume less than 10 Kbytes of data per minute of sound. This is because the MIDI file does not contain the sampled audio data; it contains only the instructions needed by a synthesizer to play the sounds.

Samples have the advantage that you as the interface designer know exactly what your sound will sound like. With MIDI you are at the mercy of the synthesizer on the user's machine. It may be of very poor quality and so the sounds might not sound anything like the ones you designed. With samples all of the information about the sound is stored so that you can guarantee it will sound like the recording you made (given the possible limitations of the speakers on the user's machine).

It is also not possible to synthesize all sounds. Much of the work in the area of sound synthesis has focused on synthesizing musical instruments. There are few synthesizers that can do a good job of creating natural, everyday sounds. If you want to use natural sounds in your interface then you are really limited to using samples.

4.4 Three dimensional sound

Much of the recorded sound we hear is in *stereo*. A stereo recording uses differences in intensity between the ears. 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 two loudspeakers or inside the head between the listeners' ears if they are wearing headphones. This simple, inexpensive technique can give useful spatial cues at the auditory interface. This is being taken further to make sounds appear as coming from around a user (in virtual 3D) when only a small number of loudspeakers (or even just a pair of headphones) are used (Begault, 1994). As well as the ITD and IID, in the real world we use our pinnae (the outer ear) to filter the sounds coming from different directions so that we know where they are coming from. To simulate sounds as coming from around the user and outside of the head when wearing headphones, sounds entering the ear are recorded by putting microphones into the ear canals of listeners. The differences between the sound at the sound source and at the eardrum are then calculated and the differences, or ‘head-related transfer functions’ (HRTFs), derived are used to create filters with which stimuli can be synthesized (Begault, 1994). This research is important as three-dimensional auditory interfaces can be created that are more natural, with sounds presented around the user as they would be in real life.

Three-dimensional sounds over headphones can be generated by most current PC soundcards. These are often used for games but not commonly for everyday interactions, but this is beginning to change (see the work on *Nomadic Radio* in Section 6.3).

The main problem with providing simulated 3D sound through headphones comes from the HRTFs used. If your ears are not like the ears of the head from which the HRTFs were generated then you are likely to feel that the sounds are coming from inside your head, and not outside. It is also very easy to confuse front and back so that listeners cannot tell if a sound is in front or behind. Vertical positioning is also difficult to do reliably. These means that many designers who use 3D sound in their interfaces limit the sounds to a plane cutting through the head horizontally at the level of the ears. This reduces the space in which sounds can be presented but avoids many of the problems of users not being able to localize the sounds properly.

5 Non-speech sound presentation techniques

Two main types of information presentation techniques have emerged in the area of sound in human-computer interfaces: *Auditory Icons* and *Earcons*. Substantial research has gone into developing both of these and the main work is reviewed below.

5.1 Auditory Icons

Gaver (1989; 1997) developed the idea of *Auditory Icons*. These are natural, everyday sounds that can be used to represent actions and objects within an interface. He defines them as “everyday sounds mapped to computer events by analogy with everyday sound-producing events. Auditory icons are like sound effects for computers". Auditory icons rely on an analogy between the everyday world and the model world of the computer (Gaver, 1997) (for more examples of the use of earcons see the work on Mercator and Audio Aura described in Sections 6.2 and 6.3 below).

Gaver uses sounds of events that are recorded from the natural environment, for example tapping or smashing sounds. He uses an 'ecological listening' approach, suggesting 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. 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.

Gaver used these ideas to create auditory icons and from these built the *SonicFinder* (Gaver, 1989). This is an interface that ran on the Apple Macintosh and provided auditory representations of some objects and actions within the interface. Files were given a wooden sound, applications a metal sound and folders a paper sound. The larger the object the deeper the sound it made. Thus, selecting an application meant tapping it - it made a metal sound which confirmed that it was an application and the deepness of the sound indicated its size. Copying used the idea of pouring liquid into a receptacle. The rising of the pitch indicated that the receptacle was getting fuller and the copy progressing.

To demonstrate how the SonicFinder worked a simple interaction is provided in Figure 3 showing the deletion of a folder. In A) a folder is selected by tapping on it, this causes a 'papery' sound indicating that the target is 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).

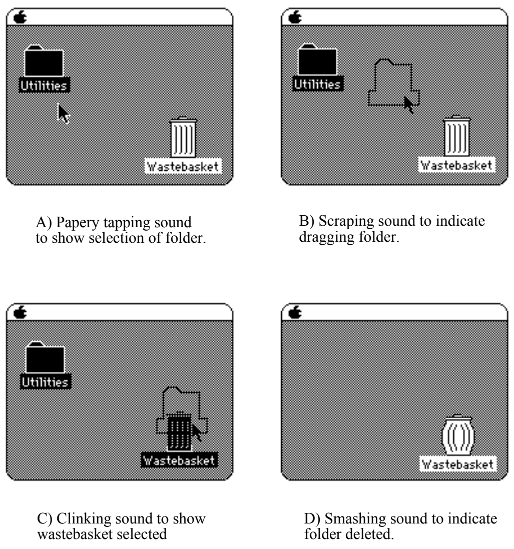
Utilities
Wastebasket
A) Papery tapping sound to show selection of folder.
B) Scraping sound to indicate dragging folder.
C) Clinking sound to show wastebasket selected
D) Smashing sound to indicate folder deleted.

Figure 3: An interaction showing the deletion of a folder in the SonicFinder (from Gaver, 1989).

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 was 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 developed the ideas from the SonicFinder further in the *ARKola* system (Gaver, Smith, & O'Shea, 1991) which modeled a soft drinks factory. The simulation consisted of a set of nine machines 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. 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 users controlled the factory, with each user able to see approximately one third of the whole plant. This form of plant was chosen because it allowed Gaver *et al.* to investigate how the sounds would effect the way users handled the given task and how they affected the way people collaborated. It was also an opportunity to investigate how different sounds would combine to form an auditory *ecology* (integrated set of sounds). Gaver *et al.* relate the way the different sounds in the factory combined to the way a car engine is perceived. Although the sounds are generated by multiple distinct components, these combine to form what is perceived as a unified sound. If something goes wrong, the sound of the engine will change, alerting the listener to the problem, but in addition, to a trained ear the change in the sound would alert the listener to the nature of the problem. The sounds used to indicate the performance of the individual components of the factory were designed to reflect the semantics of the machine.

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.

An informal evaluation was undertaken where pairs of users were observed controlling the plant, either with or without sound. These observations indicated that the sounds were effective in informing the users about the state of the plant and that the users were able to differentiate the different sounds and identify the problem when something went wrong. When the sounds were used there was much more collaboration between the two users. This was because each could hear the whole plant and therefore help out if there were problems with machines that the other was controlling. In the visual condition users were not as efficient at diagnosing what was wrong even if they knew there was a problem.

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. Problems did, however, occur with some of the warning sounds used as Gaver *et al.* indicate: "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 something had gone wrong.

5.1.1 Design guidelines for auditory icons

As yet few formal studies have been undertaken to investigate the best design for auditory icons, so there is little guidance for how to design them effectively. Mynatt (1994) has proposed the following basic methodology: (1) Choose short sounds which have a wide bandwidth, and where length, intensity, and sound quality are roughly equal. (2) Evaluate the identifiability of the auditory cues using free-form answers. (3) Evaluate the learnability of the auditory cues which are not readily identified. (4) Test possible conceptual mappings for the auditory cues using a repeated measures design where the independent variable is the concept that the cue will represent. (5) Evaluate possible sets of auditory icons for potential problems with masking, discriminability and conflicting mappings. (6) Conduct usability experiments with interfaces using the auditory icons.

5.2 Earcons

Earcons were developed by Blattner, Sumikawa & Greenberg (1989). They use abstract, synthetic tones in structured combinations to create auditory messages. Blattner *et al.* (1989) 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 Auditory Icons, there is no intuitive link between the earcon and what it represents; the link must be learned. They use a more traditional musical approach than auditory icons.

Earcons are constructed from simple building blocks called *motives* (Blattner et al., 1989). These are short, rhythmic sequences of pitches that can be combined in different ways. Blattner suggest the most important features of motives are:

Rhythm: Changing the rhythm of a motive can make it sound very different. Blattner *et al.* (1989) describe this as the most prominent characteristic of a motive.

Pitch: There are 96 different pitches in the western musical system and these can be combined to produce a large number of different motives.

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.

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).

There are two basic ways in which earcons can be constructed. The first, and simplest, are *Compound earcons*. These are simple motives that can be concatenated to create more complex earcons. For example, a set of simple, one element motives might represent various system elements such as 'create', 'destroy, 'file' and 'string' (see Figure 4A) these could then be concatenated to form earcons (Blattner et al., 1989). In the figure the earcon for 'create' is a high-pitched sound that gets louder, for 'destroy' it is a low-pitched sound that gets quieter. For 'file' there are two long notes that fall in pitch and for 'string' two short notes that rise. In Figure 4B 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 earcons.

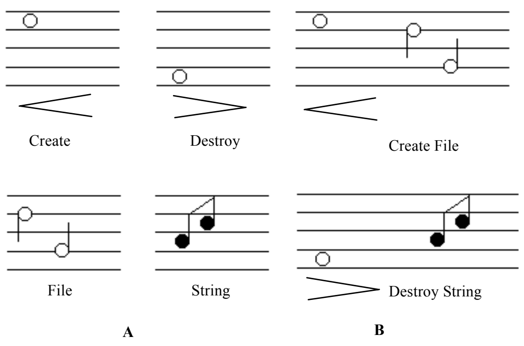

Figure 4: Compound earcons. *A* shows the four audio elements ‘create’, ‘destroy’, ‘file’ and ‘string’. *B* shows the compound earcons ‘create file’ and ‘destroy string’ (Blattner et al., 1989).

Hierarchical earcons are more complex but can be used to represent more complex structures in sound. Each earcon is a node in a tree and inherits properties from the earcons above it. Figure 5 shows a hierarchy of earcons representing a family of errors. The top level of the tree is the family rhythm. This sound just has a rhythm and no pitch; the sounds used are clicks. The rhythmic structure of level 1 is inherited by level 2 but this time a second motive is added where pitches are put to the rhythm. At this level, Blattner *et al.* suggest that the timbre should be a sine wave, which produces a ‘colorless’ sound. This is done so that at level 3 the timbre can be varied. At level 3 the pitch is also raised by a semitone to make it easier to differentiate from the pitches inherited from level 2. Other levels can be created where register and dynamics are varied.

Blattner *et al.* proposed the design of earcons but did not develop or test them. Brewster *et al.* (1994) carried out a detailed evaluation of compound and hierarchical earcons based on the design proposed by Blattner, simple system beeps and a richer design based on more complex musical timbres using psychoacoustical research (see Section 3 and Chapter 26). In these experiments participants were presented with earcons representing families of icons, menus and combinations of both (examples can be heard at www.dcs.gla.ac.uk/~stephen/demos.shtml). They heard each sound three times and then had to identify them when played back. Results showed that the more complex musical earcons were significantly more effective than both the simple beeps and Blattner's proposed design, with over 80% recalled correctly. Brewster *et al.* found that timbre was a much more important than previously suggested whereas pitch on its own was difficult to differentiate. The main design features of the earcons used were formalized into a set of design guidelines:

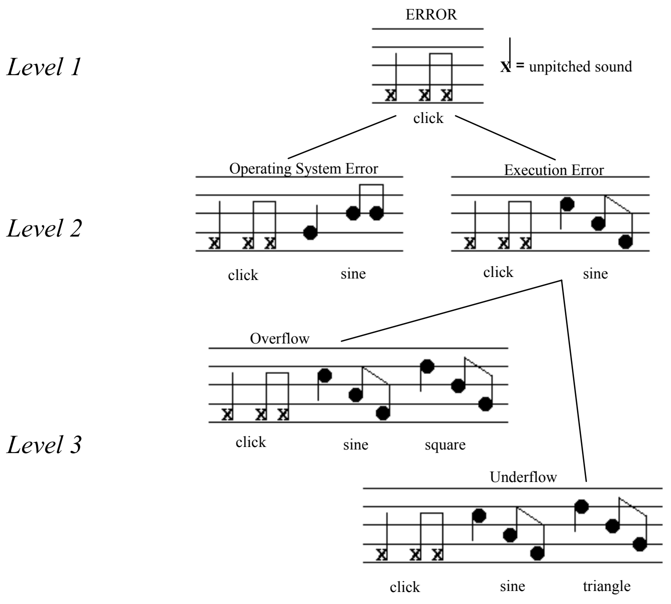

Figure 5: A hierarchy of earcons representing errors (from Blattner et al., 1989).

Timbre: This is the most important grouping factor for earcons. Use musical instrument timbres with multiple harmonics as this helps perception and can avoid masking. These timbres are more recognizable and differentiable.

Pitch and Register: If listeners are to make absolute judgments of earcons then pitch/register should not be used as a cue on its own. A combination of register and another parameter gives better rates of recall. If register alone must be used then there should be large differences (two or three octaves) between earcons. Much smaller differences can be used if relative judgments are to be made. The maximum pitch used should be no higher than 5kHz and no lower than 125Hz-150Hz so that the sounds are not easily masked and are within the hearing range of most listeners.

Rhythm, duration and tempo: Make rhythms as different as possible. Putting different numbers of notes in each earcon is very effective. Earcons are likely to be confused if the rhythms are similar even if there are large spectral differences. Very short note lengths might not be noticed so do not use very short sounds (less than 0.03 seconds). Earcons should be kept as short as possible so that they can keep up with interactions in the interface being sonified. Two earcons can be played in parallel to speed up presentation.

Intensity: This should not be used as a cue on its own because it is a major cause of annoyance. Earcons should be kept within a narrow intensity range so that annoyance can be avoided (see Section 2.4 above for more on this issue).

One aspect that Brewster also investigated was musical ability – as earcons are based on musical structures is it only musicians who can use them? The results showed that the more complex earcons were recalled equally well by non-musicians as they were by musicians, indicating that they are useful to a more general audience of users.

In a further series of experiments Brewster (1998b) looked in detail at designing hierarchical earcons to represent larger structures (with over 30 earcons at four levels). These were designed building on the guidelines above. Users were given a short training period and then were presented with sounds and they had to indicate where the sound was in the hierarchy.

Results were again good with participants recalling over 80% correctly, even with the larger hierarchy used. The study also looked at the learning and memorability of earcons over time. Results showed that even with small amounts of training users could get good recall rates, and that the recall rates of the same earcons tested a week later was unchanged. More recent studies (Leplâtre & Brewster, 2000) have begun to investigate hierarchies of over 150 nodes (for representing mobile telephone menus). For examples of earcons in use see the sonically enhanced widgets, Audiograph and Palm III work in Sections 6.1, 6.2 and 6.3.

5.3 Comparing auditory icons and earcons

Earcons and auditory icons are both effective at communicating information in sound. There is more formal evidence of this for earcons as more basic research has looked at their design. There is less basic research into the design of auditory icons but the systems that have used them in practice have been effective. More detailed research is needed into auditory icons to correct this problem and to provide designers with design guidance on how to create effective sounds. It may be that each has advantages over the other in certain circumstances and that a combination of both is the best. In some situations the intuitive nature of auditory icons may make them favorable. 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.

The advantage of auditory icons over earcons is that they are easy to learn and remember because they are based on natural sounds and the sounds contain a semantic link to the objects they represent. This may make their association to certain, more abstract, actions or objects within an interface more difficult. Problems of ambiguity can also occur when natural sounds are taken out of the natural environment and context is lost (and people may also have their own idiosyncratic mappings). If the meanings of auditory icons must be learned then they lose some of their advantages.

Earcons are abstract so their meaning must always be learned. This may be a problem, for example, in ‘walk up and use’ type applications. Research has shown that little training is needed if the sounds are well designed and structured. Leplatre and Brewster (2000) have begun to show that it is possible to learn the meanings implicitly whilst using an interface that generates the sounds as it is being used. However, some form of learning must take place. According to Blattner *et al.* (1989), earcons may have an advantage when there are many highly structured sounds in an interface. With auditory icons each one must be remembered as a distinct entity because there is no structure linking them together. With earcons there can a strong structure linking them that can easily be manipulated. There is not yet any experimental evidence to support this.

'Pure' auditory icons and earcons make up the two ends of a presentation continuum from representational to abstract (see Figure 6). In reality things are less clear. Objects or actions within an interface that do not have an auditory equivalent must have an abstract auditory icon made for them. The auditory icon then moves more towards the abstract end of the continuum. When hearing an earcon, the listener may hear and recognize a piano timbre, rhythm and pitch structure as a kind of 'catch-phrase' representing an object in the interface; he/she does not hear all the separate parts of the earcon and work out the meaning from them (listeners may also try and put their own representational meanings on earcons, even if the designer did not intend it as found by Brewster, 1998b). The earcon then moves more towards the representational side of the continuum. Therefore, earcons and icons are not necessarily as far apart as they might appear.

There are not yet any systems that use both types of sounds to their full extent and this would be an interesting area to investigate. Some parts of a system may have natural analogues in sound and therefore auditory icons could be used; other parts might be more abstract or structured and earcons would be better. The combination of the two would be the most beneficial. This is an area ripe for further research.

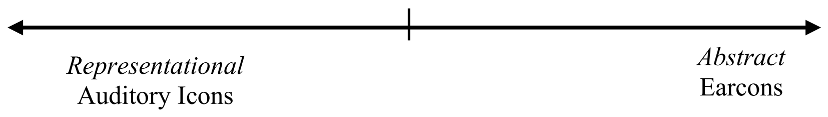

Figure 6: The presentation continuum of auditory icons and earcons.

6 The applications of auditory output

Auditory output has been used in a wide range of different situations and applications. This section will outline some of the main areas of use and will highlight some of the key papers in each area (for more uses of sound see the ICAD or ACM CHI series of conferences).

6.1 Sonic enhancement and graphical replacement in GUIs

One long-running strand of research in the area of auditory output is in the addition of sound to standard graphical displays to improve usability. One reason for doing this is that users can become overloaded with visual information on large, high-resolution displays. In highly complex graphical displays users must concentrate on one part of the display to perceive the visual feedback, so that feedback from another part may be missed. This becomes very important in situations where users must notice and deal with large amounts of dynamic data. For example, imagine you are working on your computer writing a report and are monitoring several on-going tasks such as a compilation, a print job and downloading files over the Internet. The word-processing task will take up your visual attention because you must concentrate on what you are writing. In order to check when your printout is done, the compilation has finished or the files have downloaded, you must move your visual attention away from the report and look at these other tasks. This causes the interface to intrude into the task you are trying to perform. If information about these other tasks was presented in sound then you could continue looking at the report but hear information in the background about the other tasks. To find out how the file download was progressing you could just listen to the download sound without moving your visual attention from the writing task.

There are two aspects to this work: *enhancement* and *replacement*. Research has gone into looking at how feedback from widgets can be improved by the addition of sound. Alternatively, feedback can be taken away from the visual sense altogether and just be presented in sound. Examples of both types of designs are described below.

One of the earliest pieces of work on sonic enhancement of an interface was Gaver's SonicFinder described above (1989). This used auditory icons to present information about the Macintosh interface redundantly with the graphical display.

Brewster investigated the addition of sound to enhance graphical buttons (Brewster, 1998a). An analysis of the way buttons are used was undertaken highlighting some usability problems. It was found that the existing, visual feedback was did not indicate when mis-presses of a button might have occurred. For example, the selection of a graphical button is shown in Figure 7 (starting with 1.A and 2.A). The button highlights when it is pressed down (Figure 7 1.B and 2.B). There is no difference in feedback between a correct selection (Figure 7 1.C) and a mis-selection (Figure 7 2.C), where the user moves the mouse off the graphical button before the selection is complete. The user could therefore 'slip off' the button, fail to press it and get no feedback. This error can happen when the user is moving away from the button and on to some other task. For example, the user moves to a toolbar to press the 'Bold' button and then moves back to the text to position the cursor to start typing. The button press and the mouse move overlap and the button is not pressed. It is hard for the user to notice this because no feedback is given.

The problems could not easily be solved by adding more graphical feedback: the user is no longer looking at the button's location so any feedback given there will be missed. Feedback could be given at the mouse location but we cannot be sure the user will be looking there either. Brewster designed a new button that used auditory feedback to indicate more about the state of the button. This was advantageous as sound is omni-directional and the user does not need to focus attention on any part of the screen to perceive it.

Three earcons were used to improve the effectiveness of graphical buttons. An organ timbre was used for all of the sounds. When the user moved over a button a continuous tone was played at

130Hz at a volume just above the background sound level. This informed the user the cursor was over the target but could easily be habituated. When the mouse was pressed down over the graphical button a continuous tone was played at 261Hz. The third sound indicated that the graphical button had been successfully selected. This sound consisted of two short tones with a pitch of 1046Hz and duration of 40ms. This sound was not played if a slip-off error occurred. If the user pressed the button very quickly then only the success sound was played to avoid unnecessary feedback.

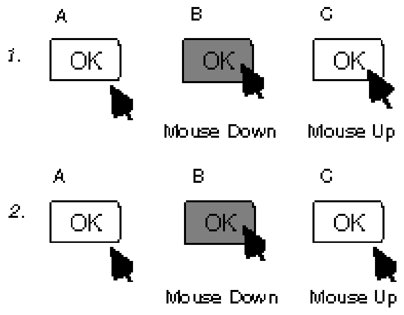

Figure 7: The visual feedback presented by a graphical button when selected. 1 shows a correct selection and 2 shows a slip-off (from Brewster, 1998a).

An experimental evaluation of these sounds was undertaken. The participants were presented with a graphical number pad of buttons. Using this number pad, they had to enter as many five digit strings as possible within 15 minutes.

Results from the study showed that users recovered from slip-off errors significantly faster and with significantly fewer mouse clicks when sounds were present in the buttons. Users also significantly preferred the buttons with sound when asked to rate subjective preference. They also did not rate the buttons as more annoying than the standard graphical ones. An interesting point to note was the use of no sound when a sound was expected could be attention grabbing. The participants could easily recognize a slip-off due to the demanding nature of the success sound *not* being played. This is important as reducing the amount of feedback presented is one way to make sure that it is not annoying.

An example of a sonified widget that replaced a visual one is the auditory progress bar (Crease & Brewster, 1998). Progress bars are a common feature of most graphical interfaces indicating the current state of a task, such as downloading documents from the Web or copying files from one disk to another. If downloads are occurring over long periods of time then the progress bar is likely to get pushed behind the window in which the user is working and completion or errors may not be noticed. A better presentation method would be to use sound so that the user could monitor the download whilst working in another window at the same time. In Crease and Brewster's progress bar four sounds were used:

Endpoint Sound: This marked the end of the progress bar. This was a single bass guitar note played for 500ms every second during the download and was of a fixed pitch (65 Hz). This sound was played as a discrete note every second rather than a continuous note in order to minimize any annoyance. A bass instrument was chosen because this sound is the root upon which the Progress sound is based, in a similar way that a bass line is the root to a melody in a tune.

Progress Sound: This was a single organ note played for 250ms every second during the download, half a second after the end point sound started. The pitch of this note was used to indicate the percentage completed. The pitch of the note starts at 261 Hz and as the task progresses this pitch moved down towards 65 Hz (the pitch of the Endpoint sound) in proportion to the amount of the task completed.

Rate Of Progress Sound: This sound was used to indicate the rate at which the task was being completed. Each note used a piano instrument with a pitch of 65 Hz and played alongside the progress sound. Each note was 10ms long. At least two notes would be played at regular intervals every second. As the rate of the task increased, more notes were played, up to a maximum of 12 per second.

Completion Sound: Once the task was completed, three chords were played. Each chord consisted of two notes played for 250ms with a pitch of 65 Hz except for the third chord that was played for 500ms. Chords were played rather than individual notes to make this sound more demanding. The final chord was lengthened to indicate completion. If the task completed unsuccessfully, a similar but discordant sound alerted the user.

An experiment was conducted to test this purely auditory progress bar against a standard visual one. Users were required to type in text whilst simulated file downloads took place (represented either visually or sonically). The results showed that subjective workload was reduced with the audio progress bar (people felt that using audio was much less demanding). People also preferred it to the standard one. They also completed their download tasks significantly faster compared to the standard visual one. These results showed that the visual progress bar could be removed from the display and users could perform better as they could share the tasks of typing and monitoring between their two senses.

Other widgets have been successfully sonified. Beaudouin-Lafon and Conversey (1996) showed that non-speech sounds could improve usability of scrollbars, Maury *et al.* (1999) added sounds to improve menu selection times in drop-down menus. Brewster and colleagues have investigated a wide range of different visual widgets and added sound to them including scrollbars, menus, tool palettes and drag and drop. These widgets have also been included into a toolkit (Crease, Gray, & Brewster, 2000) that designers can use to add sound to their interfaces (for a demonstration of some of these widgets see http://www.dcs.gla.ac.uk/~murray/audiowidgets/demos.shtml).

6.2 Sound for users with visual impairments

One of the most important uses for non-speech sound is in interfaces for people with visual disabilities (see Chapter 26 for more on the causes and problems of visual disabilities). One of the main deprivations caused by blindness is the problem of access to information. A blind person will typically use a screen-reader and a voice synthesizer to use a computer (Edwards, 1995; Raman, 1996). The screen reader extracts textual information from the computer's video memory and sends it to the speech synthesizer to speak it. This works well for text but not well for the graphical components of current user interfaces (Chapter 26 discusses this in more detail). It is surprising to find that most of the commercial applications used by blind people make very little use of non-speech sound, concentrating on synthetic speech output. This is very limiting (as discussed above) as speech is slow, can overload short-term memory and is not good for presenting certain types of information; for example it is not possible to render many types of images via speech so these can become inaccessible to blind people. One reason for the lack of use of sound has been how to employ it effectively, as Edwards (1995) says "Currently the greatest obstacle to the exploitation of the variety of communications channels now available is our lack of understanding of how to use them". The combination of speech and non-speech sounds can increase the amount of information presented to the user. As long as this is done in a way that does not overload the user then it can improve access to information. Some of the main research into the use of non-speech auditory in interfaces for blind people will now be described.

Soundtrack was an early attempt to create a word-processor designed to be used by blind persons and was developed by Edwards (1989). It used earcons and synthetic speech as output and was designed so that the objects a sighted user would see in an interface, for example menus and dialogues, were replaced by auditory equivalents that were analogies of their visual counterparts. Its interface was constructed from auditory objects with which the user could interact. They were defined by a location, a name, a sound and an action. They were arranged into a grid of two layers (see Figure 8), analogous to menus.

Each auditory object made a sound when the cursor entered it and these could be used to rapidly navigate around the screen. Soundtrack used sine waves for its audio feedback. Chords were built-up for each menu dependent on the number of menu items. For the edit menu a chord of four notes was played because there were four menu items within it (cut, copy, paste and find).

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

Figure 8: Soundtrack's main screen (from Edwards, 1989).

The base sounds 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.

Double-clicking on a menu object took the user to the second level of Soundtrack - the menu items associated with the chosen menu. Moving around is similar to moving on the main screen: each item had a tone - when moving down the menu the pitch of the tone decreased and when moving up the pitch increased.

Results from trials with blind users showed that it was very successful and users could navigate around the interface and find menus and items easily. 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 PC 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 purely auditory interfaces was possible.

The approach taken in Soundtrack was to take the visual interface to a word processor and translate it into an equivalent auditory form. The *Mercator* system (Mynatt & Edwards, 1995; Mynatt & Weber, 1994) took a broader approach. The designers' goal was to model and translate the graphical interfaces of X Windows applications into sound without modifying the applications (and thus create a more general solution than Soundtrack's). Their main motivation was to simulate many of the features of graphical interfaces to make graphical applications accessible to blind users and keep coherence between the audio and visual interfaces so that blind and sighted users could interact and work together on the same applications. This meant that the auditory version of the interface had to facilitate the same mental model as the visual one. This did not mean that they translated every pixel on the screen into an auditory form; instead they modeled the interaction objects that were present. If there was a menu, dialogue box or button on the screen then it was displayed, but in an auditory form. Modeling the pixels exactly in sound was ineffective due to the very different nature of visual and auditory media and the fact that graphical interfaces had been optimized to work with the visual sense (for example, the authors claim that an audio equivalent of overlapping windows was not needed as overlapping was just an artifact of a small visual display). Non-speech sound was an important aspect of their design to make the iconic parts of a graphical interface usable.

Mercator used three levels of non-speech auditory cues to convey symbolic information presented as icons in the visual interface. The first level addressed the question of "what is this object?" In Mercator, the type of an interface object was conveyed with an auditory icon. For example, touching a window sounded like tapping a piece of glass, container objects sounded like a wooden box with a creaky hinge and text fields used the sound of a manual typewriter. While the mapping was easy for interface components such as trashcan icons, it was less straightforward components that did not have simple referents in reality (e.g., menus or dialogue boxes, as discussed in Section 5.1). In Mercator, auditory icons were also parameterized to convey more detailed information about specific attributes such as menu length. Global attributes were also mapped into changes in the auditory icons. For example highlighting and graying-out are common to a wide range of different widgets. To represent these Mynatt *et al.* used sound filters. A low-pass filter was used to make the sound of a gray-ed object sounds duller and more muffled.

The next two pieces of work look at how visual representations of data can be transferred into sound. Mansur, Blattner, and Joy (1985) reported the first significant study of presenting data in sound. Their study, which laid out the research agenda for subsequent research in sound graphs, used sound patterns to represent two-dimensional line graphs. Their prototypical system provided blind people with a means of understanding line graphs similar to printed graphs for those with sight. Their study used auditory graphs that had a three-second continuously varying pitch to present the graphed data. The value on the y-axis of the graph was mapped to pitch and the x-axis to time, this meant that a listener could hear the graph rise and fall over time in a similar way that a sighted person could see the line rising and falling. In an experiment the auditory graphs were compared to engraved plastic tactile representations of the same data.

Results were in general very good with blind participants able to extract much information about the graphs. They found that their approach was successful in allowing distinctions to be made between straight and exponential graphs, varying monotonicity in graphs, convergence and symmetry. They did, however, find that there were difficulties in identifying secondary aspects of graphs such as the slope of the curves. They suggested that a full sound graph system should contain information for secondary aspects of the graph such as the first derivative. Their suggestion was to encode this information by adding more overtones to the sound to change the timbre. They also suggested utilizing special signal tones to indicate a graph's maxima or minima, inflection points, or discontinuities.

When comparing the auditory representations with the tactile ones, they discovered that mathematical concepts such as asymmetry, monotonicity and the slopes of lines could be determined more quickly using sound. The relative accuracy of the subjects on five sets of test questions was virtually the same (sound=84.6%, tactile=88.9%), and they stated that better performance could be expected with greater training. Several further studies have been undertaken in this area to develop this presentation technique further, most notably (Flowers & Hauer, 1992; Flowers & Hauer, 1995).

Alty and Rigas developed the Audiograph system to present diagrams and drawings to blind people (Alty & Rigas, 1998; Rigas & Alty, 1997). Their main objective was to see if music alone could be used to convey meaningful information to users about the spatial layout of objects in a simple graphical drawing tool. The following information was communicated using music in the system: (1) the current position of the cursor or the position of a graphical object, (2) the nature of a graphical object (e.g. its type, size and shape) and (3) the overall position of graphical objects using various scanning techniques. A coordinate point in the display was described using a musical mapping from coordinate value to pitch (high pitch = high coordinate value), and X and Y coordinates were distinguished by timbre (Organ and Piano respectively) generated via a MIDI synthesizer. The earcon for a coordinate pair was the X coordinate motive followed by the Y coordinate motive. This was extended to present the coordinates of shapes such as lines, circles and squares. Rigas and Alty (1997) describe a *note-sequence* technique that they used to do this: taking a start point play all of the notes between the start point and the position required. This communicates length through pitch and time. They found this method successful after experimental evaluation. For example, the horizontal line in Figure 9 would be played as a series

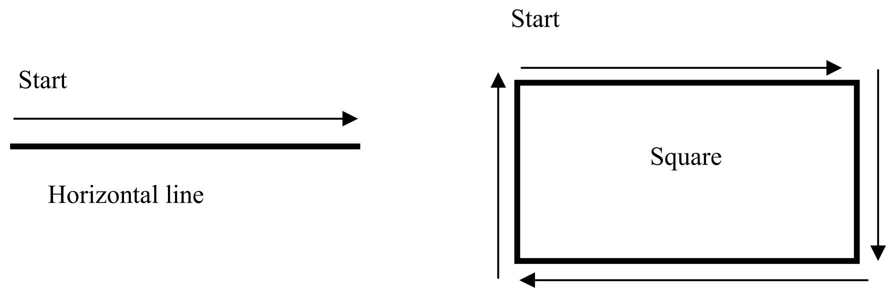

Figure 9: The musical description of graphical shapes from Rigas and Alty (1997).

of notes with increasing pitch in the organ instrument but a static pitch in the piano. The square would alternate between (rising organ, static piano), (static organ, falling piano), (falling organ, static piano) and (static organ, rising piano) as the cursor moved round from the top left.

Users navigated the drawing space using the cursor keys. Rigas and Alty discovered that even before training more than 80 percent of users recognized the shape of a circle and straight line presented with the method described. Their experiments also show that users could estimate the size of graphical objects as well as their overall shape to an accuracy of within 10%.

They also experimented with three different scanning techniques (using the note-sequence technique) to enable users to obtain an overall appreciation of the graphical space. These were: (1) Top-Down Scanning – presenting the drawing area starting at the top left hand corner and scanning progressively down the area left to right playing any object encountered along the way, (2) Center Scanning – starting at the center of the screen and scan the area in increasing circles and (3) Ascending Scanning – where scanning of objects was done in ascending order of size.

Experiments revealed that participants could get a broad picture of the diagrams with these techniques. The diagrams reproduced by their blind subjects on raised grid paper showed that the number of objects perceived in the space and their distribution were broadly in agreement but that the perception of size was not as accurate. They state that this may be partly explained by the difficulty of drawing tasks for their blind users.

These four investigations have shown that sophisticated non-speech sounds can present a lot of complex information in a very compact form. They can make something as complex as a graphical interface usable by a blind or partially sighted person.

6.3 Sound for wearable and mobile computers

One of the major growth areas in computing at the end of the 20th and start of the 21st centuries has been in mobile computing. People no longer just use computers whilst sat at a desk. Mobile telephones, Personal Digital Assistants (PDAs) and handheld computers are now some of the most widely used computers (Chapters 32 and 33 give more details). One problem with these devices is that there is a very limited amount of screen space on which to display information: the screens are small as the devices must be able to fit into the hand or pocket to be easily carried. Small screens can easily become cluttered with information and widgets and this presents a difficult challenge for interface designers (Brewster & Cryer, 1999; Sawhney & Schmandt, 2000).

The graphical techniques for designing interfaces on desktop interfaces do not apply well to handheld devices. Screen resources are limited; often screens are black and white to reduce cost and power consumption. Memory and processing power are much reduced from desktop systems. However, in many cases interface designs and interaction techniques have been taken straight from standard desktop graphical interfaces (where screen space and other resources are not a problem) and applied directly to mobile devices. This has resulted in devices that are hard to use, with small text that is hard to read, cramped graphics and little contextual information (Brewster & Murray, 2000; Hindus et al., 1995). Speech and non-speech sounds an important way of solving these problems.

Another reason for using sound is that if users are performing tasks whilst walking or driving, they cannot devote all of their visual attention to the mobile device. Visual attentional resources must remain with the main task for safety. It is therefore hard to design a visual interface that can work well under these circumstances. An alternative, sonically enhanced interface would require less visual attention and therefore potentially interfere less in the main activity in which the user is engaged.

Three main pieces of work are surveyed in this section, covering the main approaches taken in this area. The first adds sound to the existing interface of a mobile computer to improve usability; the second creates a purely auditory interface for a mobile and the third an auditory environment that users move through to receive information.

Brewster developed the ideas of sonified buttons described in Section 6.1 and applied them to buttons on the 3Com Palm series of pen-based handheld computers (Brewster & Cryer, 1999). Many of the same feedback problems with buttons apply in handhelds as in desktops, but are worse as the screen is smaller (and may be hard to see when the device is moving or the sun is shining). In addition, there is the problem of the stylus (or finger) obscuring the target on the display, which makes it difficult for users to know when they are pressing in the correct place. Simple earcons were used to overcome the problems. The same tasks were performed as described above but two different button sizes were used: standard (16x16 pixels) and small (this time 8x8 pixels). One aim of the experiment was to see if adding audio could reduce the size of the widgets so that screen space could be freed. Each condition had two 7-minute treatments: standard visual only buttons and visual plus sound buttons. Figure 10 shows the interface used in the experiment with the small buttons. Due to the limitations of the device upon which the experiment was performed (a 3Com PalmIII handheld computer with stylus input) three simple earcons were played. The standard PalmIII key click sound was played when a button was successfully selected; a higher pitched version of this sound was played when the stylus was pressed on a graphical button (in this case helping the users know when they were on a target that could be hard to see); and a lower pitched version of this sound was played when a button was mis-selected (see Section 6.1 for more on this error).

The results in general confirmed those of the previous experiment. Subjective workload in the sonically enhanced buttons of both sizes was reduced, as compared to their silent counterparts. In both conditions the addition of sound allowed the participants to enter significantly more 5-digit strings than in the corresponding silent treatment. A further experiment (Brewster, 2001) tested the same interface in a mobile environment with users entering 5-digit strings as they walked. Results from this experiment showed that participants walked further when sound was added and that small buttons with sound allowed as many strings to be entered as the large, silent buttons. The suggested reason for this was that users did not have to concentrate so much of their visual attention on the device, as much of the feedback needed was in sound, and so could look where they were going. This would therefore allow the size of items on the display to be reduced without a corresponding drop in usability.

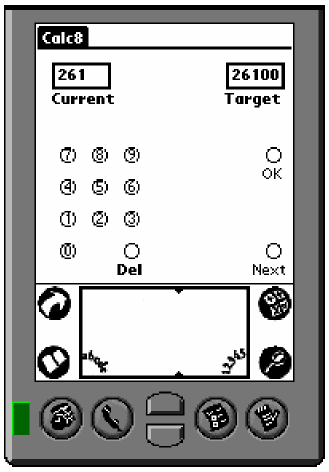

Figure 10: Screenshot of the 3Com Palm III interface used by Brewster (2001).

In a very detailed piece of work Sawhney and Schmandt (1999; 2000) developed a wearable computer-based personal messaging audio system called *Nomadic Radio* to deliver information and messages to users on the move. One of the aims of this system was to reduce the interruptions to a user caused by messages being delivered at the wrong time (for example mobile telephone calls being received in a meeting, a PDA beeping to indicate an appointment in the middle of a conversation). In the system, users wore a microphone and shoulder-mounted loudspeakers that provide a basic planar 3D audio environment (see Section 4.4) through which the audio was presented. A clock face metaphor was used with 12:00 in front of the user's nose, 3:00 by the right ear, 6:00 directly behind the head, etc. Messages were then presented in the position appropriate to the time that they arrived. The advantage of the 3D audio presentation (as described above) is that it allows users to listen to multiple sound streams at the same time and still be able to distinguish and separate each one (the ‘Cocktail party’ effect).

The system used a context-based notification strategy that dynamically selected the appropriate notification method based on the user’s attentional focus. Seven levels of auditory presentation were used from silent to full speech rendering. If the user was engaged in a task then the system was silent and no notification of an incoming call or message would be given (so as not to cause an interruption). The next level used ‘ambient’ cues (based on auditory icons) with sounds like running water indicating that the system was operational. These cues were designed to be easily habituated but to let the user know that the system was working. The next level was a more detailed form of auditory cue giving information on system events, task completions and mode transitions. For example, a ringing telephone sound was used to indicate the arrival of voicemail. These were more attention grabbing than the ambient cues and would only be played if the user was not fully occupied. The next four levels of cue used speech, expanding from a simple message summary up to the full text of a voicemail message. These might be used if the person wearing nomadic radio was not involved in tasks that required detailed attention. The system attempted to work out the appropriate level to deliver the notifications by listening to the background audio level in the vicinity of the user (using the built-in microphone) and if the user was speaking or not. For example, if the user was speaking the system might use an ambient cue so as not to interrupt the conversation. Users could also press a button on the device to indicate they were busy and so turn it to silent.

Audio Aura was created by Mynatt *et al.* (1997; 1998) “to provide serendipitous information, via background auditory cues, that is tied to people’s physical actions in the workplace”. In a similar way to Nomadic Radio, Audio Aura used auditory icons to provide background information that did not distract users.

The system used active badges so that the location of users could be identified and appropriate audio cues given, along with wireless headphones so that users could hear the sounds without distracting others. The location information from the active badges was combined with other data sources such as on-line calendars and email. Changes in this information triggered audio cues sent to the user through the headphones. Audio Aura is an audio equivalent to the ideas of ambient computing developed by Ishii (1997). Here, similar information is presented via visual cues in the physical environment.

Here are some examples of how the system might be used. In the first the user might go the office coffee room and as he/she enters the room hear information about the number and type of email messages currently waiting. This would give the user a cue as to whether or not to stay and talk to colleagues or go back to the office to answer the messages. In the second example a user goes to a colleague’s office but the occupant is not there. Audio Aura would play sounds indicating if the occupant has been in recently or away for a longer period of time. Members of work groups who are physically separated can find it hard to get a sense of group activity levels. Data was collected about whether workgroup members were in the office, working on shared documents or in meetings. Audio Aura could then use this to present a ‘group pulse’ to all of the members of the group so that they knew what was going on.

The authors were keen to make sure the sounds were not distracting and attention grabbing – they were meant to give background information and not to be alarms. To this end great care was taken with the cue design. They attempted to design ‘sonic ecologies’ – groups of sounds that fitted together into a coherent whole. For example, one set of cues was based on a beach scene. The amount of new email was mapped to seagull cries: the more mail the more the gulls cried. Group activity levels were mapped to the sound of surf: the more activity going on within the group the more active the waves became. These cues were very subtle and did not grab users’ attention but some learning of the sounds would be needed as they are quite abstract.

One novel aspect of Audio Aura was the prototyping technique the designers used. They built initial systems using VRML (a 3D modeling language which includes 3D audio – www.vrml.org). This allowed them to simulate their office building and then try out a range of audio cue designs before integrating all of the physical technology needed for the real system. This allowed them to test the positioning of sensors, for example, to ensure that they were correctly placed before putting them into the real office environment.

7 Conclusions

Research into the use of non-speech sounds for output at the human-computer interface really began in the early 1990's and there has been rapid growth since then. It has shown its benefits in a wide range of different applications from systems for blind people to wearable computers. There are many good examples that designers can look at to see how sounds can be used effectively and design guidelines are now starting to appear. Two areas are likely to be key in its future growth. The first is in combining it with speech to make the most of the advantages of both. This is an area ripe for further investigation and there are lots of interesting interaction problems that can be tackled when they are both used together. The second area in which non-speech sound has a large part to play in the near future is in mobile computing devices. The number of these is increasing very rapidly but they are hard to use because the screens are small. This is exactly the situation where sound has many advantages – it does not take up any precious screen space and users can hear it even if they cannot look at their device. This is an area that interface designers can really get some major usability improvements if they use non-speech auditory output.

8 References

Aldrich, F. K., & Parkin, A. J. (1989). Listening at speed. British journal of visual impairment and blindness, 7(1), 16-18.

Alty, J., & Rigas, D. (1998). Communicating graphical information to blind users using music: the role of context. Paper presented at the Proceedings of ACM CHI'98, Los Angeles, CA.

Alty, J. L. (1995). Can we use music in human-computer interaction? Paper presented at the Proceedings of HCI'95, Huddersfield, UK.

Baddeley, A. (1990). Human Memory: Theory and Practice. London: Lawrence Erlbaum Associates.

Bagwell, C. (1998, 14/11/1998). Audio file formats FAQ [Web]. Bagwell, C. Retrieved April, 2001, from the World Wide Web: http://home.sprynet.com/%7Ecbagwell/AudioFormats.html

Barker, P. G., & Manji, K. A. (1989). Pictorial dialogue methods. International Journal of Man-Machine Studies, 31, 323-347.

Beaudouin-Lafon, M., & Conversy, S. (1996). Auditory illusions for audio feedback. Paper presented at the ACM CHI'96 Conference Companion, Vancouver, Canada.

Begault, D. R. (1994). 3-D sound for virtual reality and multimedia. Cambridge, MA: Academic Press.

Berglund, B., Harder, K., & Preis, A. (1994). Annoyance perception of sound and information extraction. Journal of the Acoustical Society of America, 95(3), 1501-1509.

Blattner, M., & Dannenberg, R. B. (Eds.). (1992). Multimedia Interface Design. New York: ACM Press, Addison-Wesley.

Blattner, M., Sumikawa, D., & Greenberg, R. (1989). Earcons and icons: Their structure and common design principles. Human Computer Interaction, 4(1), 11-44.

Bly, S. (1982). Sound and computer information presentation (Unpublished PhD Thesis UCRL53282): Lawrence Livermore National Laboratory.

Brewster, S. A. (1998a). The design of sonically-enhanced widgets. Interacting with Computers, 11(2), 211-235.

Brewster, S. A. (1998b). Using Non-Speech Sounds to Provide Navigation Cues. ACM Transactions on Computer-Human Interaction, 5(3), 224-259.

Brewster, S. A. (2001). Overcoming the Lack of Screen Space on Mobile Computers. Accepted for publication in Personal and Ubiquitous Technologies, 5.

Brewster, S. A., & Cryer, P. G. (1999). Maximising Screen-Space on Mobile Computing Devices. Paper presented at the Summary Proceedings of ACM CHI'99, Pittsburgh, PA.

Brewster, S. A., & Murray, R. (2000). Presenting dynamic information on mobile computers. Personal Technologies, 4(4), 209-212.

Brewster, S. A., Wright, P. C., & Edwards, A. D. N. (1994). A detailed investigation into the effectiveness of earcons. Paper presented at the Proceedings of ICAD'92, Santa Fe Institute, Santa Fe.

Buxton, W. (1989). Introduction to this special issue on nonspeech audio. Human Computer Interaction, 4(1), 1-9.

Buxton, W., Gaver, W., & Bly, S. (1991). Tutorial number 8: The use of non-speech audio at the interface. Paper presented at the Proceedings of ACM CHI'91, New Orleans.

Chowning, J. (1975). Synthesis of complex audio spectra by means of frequency modulation. Journal of the Audio Engineering Society, 21(7), 526-534.

Crease, M., Gray, P., & Brewster, S. A. (2000). Caring, Sharing Widgets. Paper presented at the Proceedings of BCS HCI2000, Sunderland, UK.

Crease, M. C., & Brewster, S. A. (1998). Making progress with sounds - The design and evaluation of an audio progress bar. Paper presented at the Proceedings of ICAD'98, Glasgow, UK.

Edwards, A. D. N. (1989). Soundtrack: An auditory interface for blind users. Human Computer Interaction, 4(1), 45-66.

Edwards, A. D. N. (Ed.). (1995). Extra-Ordinary Human-Computer Interaction. Cambridge, UK: Cambridge University Press.

Edworthy, J., Loxley, S., & Dennis, I. (1991). Improving auditory warning design: Relationships between warning sound parameters and perceived urgency. Human Factors, 33(2), 205-231.

Flowers, J. H., & Hauer, T. A. (1992). The ear's versus the eye's potential to assess characteristics of numeric data: Are we too visuocentric? Behavior Research Methods, Instruments, and Computers, 24, 258-264.

Flowers, J. H., & Hauer, T. A. (1995). Musical versus visual graphs: cross-modal equivalence in perception of time series data. Human Factors, 37, 553-569.

Frysinger, S. P. (1990). Applied research in auditory data representation. Paper presented at the Extracting meaning from complex data: processing, display, interaction. Proceedings of the SPIE/SPSE symposium on electronic imaging., Springfield, VA.

Gaver, W. (1989). The SonicFinder: An interface that uses auditory icons. Human Computer Interaction, 4(1), 67-94.

Gaver, W. (1997). Auditory Interfaces. In M. Helander & T. Landauer & P. Prabhu (Eds.), Handbook of Human-Computer Interaction (2nd ed., pp. 1003-1042). Amsterdam: Elsevier.

Gaver, W., Smith, R., & O'Shea, T. (1991). Effective sounds in complex systems: The ARKola simulation. Paper presented at the Proceedings of ACM CHI'91, New Orleans.

Gelfand, S. A. (1981). Hearing: An introduction to psychological and physiological acoustics. New York: Marcel Dekker Inc.

Heckroth, J. (1994). A Tutorial on MIDI and Wavetable Music Synthesis [Web]. Crystal Semiconductor. Retrieved April, 2001, from the World Wide Web: http://kingfisher.cms.shu.ac.uk/midi/main_p.htm

Hindus, D., Arons, B., Stifelman, L., Gaver, W., Mynatt, E., & Back, M. (1995). Designing auditory interactions for PDAs. Paper presented at the Proceedings of the 8th ACM symposium on User interface and software technology.

Ishii, H., & Ulmer, B. (1997). Tangible Bits: towards seamless interfaces between people, bits and atoms. Paper presented at the Proceedings of ACM CHI'97, Atlanta, GA.

Kramer, G. (Ed.). (1994a). Auditory Display (Vol. Proceedings volume XVIII). Reading, MA: Addison-Wesley.

Kramer, G. (1994b). An introduction to auditory display. In G. Kramer (Ed.), Auditory Display (pp. 1-77). Reading, MA: Addison-Wesley.

Kramer, G., & Walker, B. (Eds.). (1999). Sonification Report: Status of the field and research agenda. Santa Fe: The International Community for Auditory Display.

Leplâtre, G., & Brewster, S. A. (2000). Designing Non-Speech Sounds to Support Navigation in Mobile Phone Menus. Paper presented at the Proceedings of ICAD2000, Atlanta, USA.

Mansur, D. L., Blattner, M., & Joy, K. (1985). Sound-Graphs: A numerical data analysis method for the blind. Journal of Medical Systems, 9, 163-174.

Maury, S., Athenes, S., & Chatty, S. (1999). Rhythmic menus: toward interaction based on rhythm. Paper presented at the Extended Abstracts of ACM CHI'99, Pittsburgh, PA.

McCormick, E. J., & Sanders, M. S. (1982). Human factors in engineering and design (5th ed.): McGraw-Hill.

Moore, B. C. (1997). An Introduction to the Psychology of Hearing (4th ed.). London: Academic Press.

Mynatt, E., Back, M., Want, R., & Fredrick, R. (1997). Audio aura: light-weight audio augmented reality. Paper presented at the Proceedings of ACM UIST'97, Banff, Canada.

Mynatt, E., & Edwards, K. (1995). Chapter 10: Metaphors for non-visual computing. In A. D. N. Edwards (Ed.), Extra-Ordinary Human-Computer Interaction (pp. 201-220). Cambridge, UK: Cambridge University Press.

Mynatt, E. D. (1994). Designing with auditory icons: how well do we identify auditory cues? Paper presented at the Proceedings of the CHI '94 conference companion, Boston, MA.

Mynatt, E. D., Back, M., Want, R., Baer, M., & Ellis, J. B. (1998). Designing audio aura. Paper presented at the Proceedings of ACM CHI'98, Los Angeles, CA.

Mynatt, E. D., & Weber, G. (1994). Nonvisual Presentation of Graphical User Interfaces: Contrasting Two Approaches. Paper presented at the Proceedings of ACM CHI'94, Boston, Ma.

Patterson, R. D. (1982). Guidelines for auditory warning systems on civil aircraft (CAA Paper 82017): Civil Aviation Authority, London.

Patterson, R. D. (1989). Guidelines for the design of auditory warning sounds. Proceeding of the Institute of Acoustics, Spring Conference, 11(5), 17-24.

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), 389-400.

Pohlmann, K. (1995). Principles of digital audio (3rd ed.): McGraw-Hill.

Raman, T. V. (1996). Emacspeak - A speech interface. Paper presented at the Proceedings of ACM CHI'96, Vancouver, Canada.

Rayner, K., & Pollatsek, A. (1989). The Psychology of Reading. Englewood Cliffs, New Jersey: Prentice-Hall International, Inc.

Rigas, D. I., & Alty, J. L. (1997). The use of music in a graphical interface for the visually impaired. Paper presented at the Proceedings of IFIP Interact'97, Sydney, Australia.

Roads, C. (1996). The computer music tutorial. Cambridge, MA: MIT Press.

Sawhney, N., & Schmandt, C. (1999). Nomadic radio: Scalable and contextual notification for wearable messaging. Paper presented at the Proceedings of ACM CHI'99, Pittsburgh, PA.

Sawhney, N., & Schmandt, C. (2000). Nomadic Radio: speech and audio interaction for contextual messaging in nomadic environments. ACM Transactions on Human-Computer Interaction, 7(3), 353-383.

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), 267-275.

Thimbleby, H. (1990). User Interface Design. New York: ACM Press, Addison-Wesley.