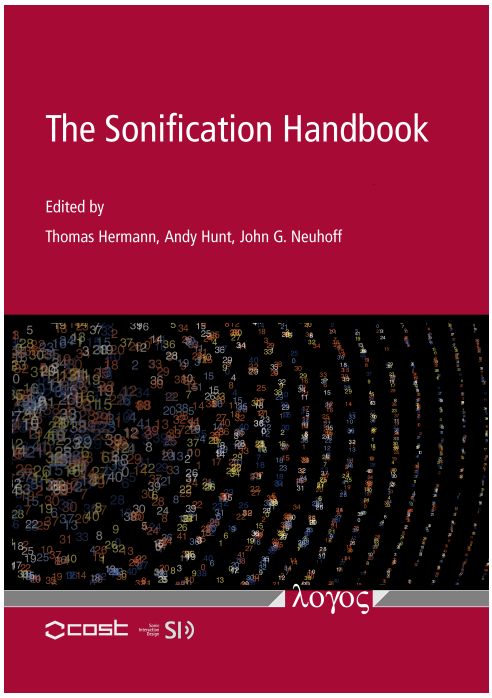


The Sonification Handbook

Edited by
Thomas Hermann, Andy Hunt, John G. Neuhoff

Logos Publishing House, Berlin, Germany
ISBN 978-3-8325-2819-5
2011, 586 pages

Online: http://sonification.de/handbook
Order: http://www.logos-verlag.com

Reference:
Hermann, T., Hunt, A., Neuhoff, J. G., editors (2011). *The Sonification Handbook*. Logos Publishing House, Berlin, Germany.

Chapter 18

Sonification for Process Monitoring

Paul Vickers

This chapter looks at a range of auditory display and sonification applications that have tackled the problem of monitoring real-time data streams and concludes with some recommendations that further research should be informed by semiotic and aesthetic thinking and should explore the use of soundscapes, steganographic embedding, model-based sonification, and spatialization as profitable techniques for sonifying monitoring data.

Reference:
Vickers, P. (2011). Sonification for process monitoring. In Hermann, T., Hunt, A., Neuhoff, J. G., editors, The Sonification Handbook, chapter 18, pages 455–491. Logos Publishing House, Berlin, Germany.

Media examples: http://sonification.de/handbook/chapters/chapter18

Chapter 18

Sonification for Process Monitoring

Paul Vickers

One of the tensions in auditory display is that sound is a temporal medium and so it is not always immediately apparent how one should approach the representation of data given that, prior to sonification, all data visualization used visual representations which rely on spatial organization. However, on the face of it, process monitoring would seem to be one of those tasks that was born to be sonified: at its heart is one or more series of temporally-related data. Monitoring entails the observation, supervision, and control of the process or system in question. To do this requires being attentive to changes in the state or behavior of the system over time so that appropriate interventions or other process-related activities may be carried out. Another feature of process monitoring is that it often has to be done as a background or secondary task or, perhaps, in parallel with one or more other primary tasks. This chapter looks at a range of auditory display and sonification applications that have tackled the problem of monitoring real-time data streams and concludes with some recommendations that further research should be informed by semiotic and aesthetic thinking and should explore the use of soundscapes, steganographic embedding, model-based sonification, and spatialization as profitable techniques for sonifying monitoring data.

18.1 Types of monitoring — basic categories

Because monitoring can be a primary or a secondary task we can classify monitoring activities into three categories: *direct*, *peripheral*, and *serendipitous-peripheral*. In a direct monitoring task we are directly engaged with the system being monitored and our attention is focused on the system as we take note of its state. In a peripheral monitoring task, our primary focus is elsewhere, our attention being diverted to the monitored system either on our own volition at intervals by scanning the system (what Jackson [66] would call a *state-vector inspection*) or through being interrupted by an exceptional event signaled by the system itself or through some monitor (such as an alarm).

In discussion of such peripheral information displays Maglio and Campbell [81, p. 241] iden-

tified that "the challenge is to create information displays that maximize information delivery while at the same time minimize intrusiveness or distraction." In Maglio's and Campbell's terms peripheral information is regarded as nonessential but nevertheless important. This, of course, is a question of context; if a network administrator is engaged in a primary task of, say, reconfiguring the firewall, then the monitoring of the network's behavior is peripheral and the information is not essential *to the task of firewall configuration* but it is, nevertheless essential to the overall goal of maintaining a healthy and stable network.

To distinguish between peripheral information that is important or essential to the overall goal but not to the immediate task in hand and that which is always only of secondary importance we can use Mynatt et al.'s [85] concept of *serendipitous* information. This is information that is useful and appreciated but not strictly required or vital either to the task in hand or the overall goal, hence the third class of process monitoring: *serendipitous-peripheral*.

Direct monitoring requires the user to be focused on the process under consideration, and so we can think of this as an information *pull* scenario: the user chooses to extract information from the interface. Peripheral monitoring is thus an information *push*: here the user is engaged in another task (or aspect of a process-related task) and it is up to the system to push significant information to the user. Serendipitous peripheral-monitoring, because the information is not vital and can be ignored by the user, is more of an information *nudge* situation: the system makes the information available and lets you know it's there, but doesn't press the matter. To summarize, we may think of three different modes of auditory process monitoring and their corresponding information push/pull type:

1. **Direct** (PULL) — the information to be monitored is the main focus of attention.
2. **Peripheral** (PUSH) — attention is focused on a primary task whilst required information relating to another task or goal is presented on a peripheral display and is monitored indirectly.
3. **Serendipitous-peripheral** (or just serendipitous for short) (PUSH/NUDGE) — attention is focused on a primary task whilst information that is useful but not required is presented on a peripheral display and is monitored indirectly.

Whilst visual displays are well suited to direct monitoring tasks (because our visual attention is focused on the display), they may not be so effective in peripheral or serendipitous-peripheral monitoring situations. It is in the monitoring of peripheral information that auditory displays come into their own, for the human auditory system does not need a directional fix on a sound source in order to perceive its presence. Tran and Mynatt [113] described sonic environmental monitors as "an extra set of eyes and ears" whilst Jenkins [69] discussed ways in which the auditory channel may be very useful for keeping users informed about background activities without being disruptive. Weiser and Brown [127] talk of 'calm computing' in which technology remains invisible in the background until needed [70].

In any case, sonification would appear to be a plausible part of the solution to the problem of monitoring processes as it allows visual attention to be focused elsewhere (perhaps on other tasks). Mountford and Gaver [84] described the difference between sound and vision in spatio-temporal terms: sound is a temporal medium with spatial characteristics whilst vision is primarily spatial in nature but with temporal features. The implication of this is that sound is good for communicating information that changes over time and can be heard over a range of spatial locations whereas vision is not so ephemeral but can only be observed at

specific locations [111]. Audio is also good at orienting, or directing, a listener to key data [71] & [77, p. 57]. Schmandt and Vallejo [101] and Vallejo [114] argued that in peripheral monitoring activities audio is *more* useful than video as well as being less intrusive.[1] Mynatt et al. [85, p. 568] went as far as to say that non-speech audio is a natural medium for creating peripheral displays, and that speech requires more attention than non-speech audio.

Even in direct monitoring situations if the visual display is very complex or crowded, or many variables need to be monitored, an auditory display provides a useful complement to (and sometimes, a replacement for) a visual display. Several related variables can be mapped to the different parameters of a single sound source allowing information-rich monitors to be built with a minimum bandwidth. Consider Fitch and Kramer's [46] sonification of heart rate, respiratory rate, blood CO_2 & O_2 levels, pupilary reflex, atrio-ventricular dissociation, fibrillation, and temperature for the monitoring of patients during surgical operations. Through careful manipulation of pitch (frequency), timbre (spectral composition), and tempo, all these variables could be monitored using only three distinct timbres. Whilst a visual display could provide precise numeric values for these variables, the sonification allowed complex monitoring to take place without the need to focus visual attention on a single-display. Smith, Pickett, and Williams [103] discussed some of the common ways such multivariate monitoring is carried out together with a framework for empirical evaluation.

18.2 Modes of Listening

One factor that must be taken into account is the difference between hearing and listening. The distinction can be simply illustrated by thinking of hearing as a push activity (a sound is projected into our attention space) and listening as a pull activity (a listener deliberately attends to an audio stream in order to identify salient characteristics or extract information/meaning). In looking for easily understood mappings for his SonicFinder system, Gaver proposed a theory of *everyday listening* [53]. This says that in everyday situations people are more aware of the attributes of the source of a sound than the attributes of the sounds themselves: it is the size of the object making the sound, the type of the object, the material it is made of, etc. that interests the everyday listener. According to the theory we hear *big* lorries, *small* children, *plastic* cups being dropped, *glass* bottles breaking, and so on. Everyday listening is in contrast to what Gaver calls *musical listening* in which we are more interested in the sensations or attributes of the sounds themselves, their pitch, their intensity, and so on. Gaver summarizes thus:

> If the fundamental attributes of the sound wave itself are of concern, the experience will be one of musical listening. Another way to say this is that everyday listening involves attending to the *distal stimuli*, while musical listening involves attending to the *proximal stimuli*. [52, p. 4]

Note that these two types of listening are not categorical — they would seem more to be points along a continuum describing general listening approaches but sharing attributes. For example, in musical listening, though one may be primarily interested in the harmonic

[1] Here, intrusiveness refers to intrusion into the environment being monitored (a domestic setting, in this case) rather than intrusion into the environment of the monitor. We can think of it in terms of telephones: a regular audio-only phone is less intrusive than a video phone.

and temporal relationships of sounds, the individual timbres and their sources are also an important part of the experience. There are 'big' sounds and 'little' sounds even in music.[2]

Gaver's classification is in a similar vein to Pierre Schaeffer's [99] *reduced listening* and Michel Chion's [30] *causal* and *semantic* modes of listening.[3] Reduced listening is the opposite of Gaver's everyday listening. In reduced listening we listen "to the sound for its own sake, as a sound object by removing its real or supposed source and the meaning it may convey".[4] Causal listening is similar to Gaver's everyday listening in that the goal is to listen to a sound in order to infer information about its source. The difference lies in the modes' intentionalities: in causal listening we deliberately seek out source information whereas in everyday listening we are aware on a more subconscious level of the origin of a sound. In a sense, Chion's causal listening is truly a mode of *listening* whereas Gaver's everday listening is more of a response to *hearing*. The third mode, semantic listening, is one in which we are not interested in the properties of a sound *per se* but in the sound's code and, by extension, its semantic or linguistic meaning. For example, semantic listening enables us to interpret the message correctly even when given by two different speakers with different pronunciations.

Gaver's work (*v.i.*) is strongly motivated by a desire to maintain good acoustic ecology. R. Murray Schafer brought acoustic ecology to the fore with his *World Soundscape Project* [100]. Schafer saw the world around us as containing ecologies of sounds. Each soundscape possesses its own ecology and sounds from outside the soundscape are noticeable as not belonging to the ecology. In Schafer's worldview we are exhorted to treat the environments in which we find ourselves as musical compositions. By this we are transformed from being mere hearers of sound into active and analytic listeners — exactly the characteristic needed to benefit most from an auditory display. When the environment produces noises that result from data and events in the environment (or some system of interest) then we are able to monitor by listening rather than just viewing. This acoustic ecology viewpoint cuts across any possible 'everyday listening' vs. 'musical listening' dichotomy as Schafer sees the everyday world as a musical composition. In so doing he brings together into a single experience Gaver's distinct acts of everyday and musical listening — we attend to the attributes of the sounds and the attributes of the sound sources equally. This is facilitated by direct relationship between the physical properties of the source and the attributes of the sounds it produces. In a sonification there is always at least one level of mapping between the source data and the sound. Furthermore, the attributes of the sounds may be quite arbitrary and have no natural relationship to the data (as will be the case in metaphor-based sonification mappings) and so the question arises as to whether such equal attention is still possible.[5] In the natural world, two unfamiliar sounds emanating from two close-together sources might also lead to confusion in the listener over which attributes belong to what source: until the source and its range of sounds is learned there is room for ambiguity. The same, it could be argued, applies to sonification. The difference is merely that not all sonifications do build upon a lifetime of learned environmental sonic associations, but as long as the mappings are learnable, equal attention ought to be possible.

[2] For example, consider Phil Spector's *Wall of Sound* production technique which led to a very dense and rich sonic tapestry that sounded 'bigger' than other recordings of the time.

[3] Also see Chapter 7 which discusses the modes of listening within the context of sonification design and aesthetics.

[4] http://www.ears.dmu.ac.uk

[5] See also the discussion of mappings and metaphors in Chapter 7 section 7.2.2.

This way of approaching sonification has started to attract interest (for instance, influencing the discussions at the First International Symposium on Auditory Graphs (AGS2005) in Limerick, Ireland in July 2005.[6] Indeed, Hermann's work on model-based interactive sonification [60] has exploited the very characteristics of everyday and musical listening by mapping data to a physical sound model (e.g., a resonator) and allowing the user to perturb the model in order to infer meaning from the data (see also Chapters 11 and 16 for discussions of developments in this field). In model-based sonification the data become an instrument which the user plays in order to study its properties.

The sections that follow will discuss, first of all, some of the main examples of auditory process monitoring across a range of application domains. After that some of the higher-level issues that arise from auditory process monitoring such as intrusion, disturbance, annoyance, aesthetic considerations, and so forth are considered.

18.3 Environmental awareness (workspaces and living spaces)

18.3.1 'Industrial' monitoring

ARKola

Drawing lessons from acoustic ecology, Gaver, Smith, and O'Shea [54] built one of the first auditory process monitoring demonstrations, the ARKola system, an auditory monitoring system for a simulated soft drink bottling plant. The system was built using the SharedARK [102] virtual physics laboratory environment and comprised nine interconnected machines each carrying out a different function in the soft drink manufacturing and bottling process. Each machine communicated its state over time using auditory icons (see Chapter 13). The choice of auditory icons was analogic so as to create a complete ecology of bottling plant sounds. For example, heating machines made a sound like a blowtorch, bottle dispensers emitted sounds of clanking bottles, and so forth. Sounds were added to communicate additional information (such as a liquid splash sound to signal when materials were being wasted; the sound of breaking glass to signify bottles being lost). The ecology of the system was especially important given that as many as fourteen sounds could be played at once. Gaver et al. had to design the auditory icons carefully so as to avoid perceptual masking effects.

In a similar study Rauterberg and Styger [91] built a simulation of an assembly line of computer numeric control (CNC) robots. In one version of the system only visual feedback was provided whilst a second system was augmented with auditory feedback. Following Gaver et al.'s ARKola approach, Rauterberg and Styger created auditory icons and arranged their sonic spectra such that perceptual masking would not occur. However, whereas ARKola supported up to fourteen concurrent sounds the CNC simulator allowed for up to thirty-eight sounds at any one time. In an experiment it was found that users who had only the visual feedback needed to move to the system control station and request status reports much more frequently than those who had the benefit of the auditory icons. Like Gaver et al. [54] and Cohen [31, 32] Rauterberg and Styger concluded that for simulations of real-world situations the use of analogic (or even metaphoric) real-world sounds is appropriate and

[6] See http://sonify.psych.gatech.edu/ags2005/

reduces the possibility of annoyance that Gaver et al. warned could arise when using 'musical' messages.[7]

ShareMon

Acting on Buxton's observation that we monitor multiple real-world background activities by their associated sounds [26], Cohen constructed the ShareMon system [32] which notified users of file sharing activity on an Apple Macintosh network. Reactions to the system were varied and strong, and Cohen describes some users finding the band-filtered pink noise that was used to represent the percentage of CPU time consumed to be "obnoxious" [32, p. 514], despite them understanding the correlation between CPU load and the pitch of the noise. On the other hand, some users described the wave sounds used in the system as soothing. Cohen extended ShareMon with an ecoustic ecology of sounds (Cohen called it a *genre*) that comprised sounds from the original Star Trek television series [31]. The sounds were mapped to various system events and users were able to successfully monitor the system events.

18.3.2 Weather monitoring

Hermann, Drees, and Ritter [59] took a nine-dimensional set of weather data and created a set of weather *prototypes*, such as "stormy winter day with snow" (sound example **S18.1**) or "hot and humid summer day" (sound example **S18.2**). The weather data were grouped into regions and individual weather vectors then allocated to one of the weather prototypes. These prototypes were then rendered in sound using multiple auditory streams to represent each aspect of the weather forecast. For example, temperature was mapped to tuned percussion timbres, rainfall to a rain sound, and so forth. The resultant sonifications were played as auditory non-speech weather reports over a radio station. Whilst not true process monitoring in the sense that it does not take place continuously and/or in real time, and each twelve-second sonification represented twenty-four hours of data, it does constitute a regular information pull scenario as listeners have to tune in to the broadcasts at set times to hear the sonification, and so serves as an instructive example for our purposes.

More recently, Bakker, van den Hoven, and Eggen [3] used auditory icons in a regular information-push scenario to play weather forecasts every thirty minutes in a shared work space. Participants in a study reported that they did not find the auditory icons distracting and after three weeks they found they noticed the auditory icons less than at the start. Although the system used a push design, participants who had no particular interest in the weather forecast did not find the sonifications annoying or distracting. The success of the system seems to have been due to a very careful sound design.

18.3.3 Home and shared work environments

ListenIN

Motivated by the desire to be able to monitor elderly relatives in their own homes remotely but still maintaining the relatives' privacy, Schmandt and Vallejo [101] built the ListenIN system. ListenIN has modules for a wide range of domestic situations, including detecting crying babies, and so is not restricted to the monitoring of older people. The ListenIN server

[7] Gaver's claim that auditory icons have less potential for annoyance than musical messages raises the question of role that aesthetics should play in sonification design. This is explored briefly at the end of this chapter (see section 18.7.2) and is discussed in more detail in Chapter 7 of this volume.

sits in the home being monitored and it determines what acoustic information is sent off the premises to the remote monitor clients. The principle of the system is that it identifies and classifies domestic noises and, if a sound matches one that the monitor has chosen to be reported, the server sends either a direct representation of the sound to the client (e.g., the sound of a baby crying for when a baby cry is detected) or a garbled version of the sound to protect privacy (e.g., garbled speech which is still recognizable as speech but in which no individual words, only the speed and intonation, can be made out). As yet, no formal evaluation has been reported. This is an example of peripheral monitoring as the sounds are not being produced at regular intervals and is akin to a sophisticated alarm.

Music Monitor

Tran and Mynatt [113] also built a system based on the user's own musical preferences for monitoring a domestic environment. Their Music Monitor used musical profiles to represent key information about the domestic activity being monitored. The intention was to provide an "extra set of eyes and ears" which could survey activities around the house and relay information about the activity states as real-time ambient music. Tran and Mynatt deliberately built the system to be what would here be classified as a serendipitous-peripheral monitoring application, the information it provides being interesting and useful but not of vital importance. In their thinking, vital information is left to alarms. In a similar approach to that of Barra et al.'s WebMelody [6] (*v.i.*), Music Monitor overlays the system information as a set of earcons onto a music track chosen by the user. Fishwick [43, 44] espouses such personalisation as an important principle of *aesthetic computing* [45]. In the realm of musical sonification it is, perhaps, even more important to cater for preference as individual musical tastes and cultural backgrounds vary a great deal. To allow the user to control the *style* of music is an important step forward in sonification practice. In a small experiment, participants were able to monitor activity state transitions displayed by Music Monitor, although situations in which there was a high level of background noise (such as cooking in the kitchen) required participants to stop what they were doing to attend to state changes played by the system. Other people in the environment who had not been told about the information display generated by Music Monitor reported only being aware of some pleasant background music — they were completely unaware of the information content in the signal.

WISP

Kilander and Lönnqvist [74, 75] sought to provide peripheral auditory monitoring through the construction of soundcapes. Their soundscapes were designed to be "weakly intrusive" to minimize the impact of the sonification on the listening environment. Kilander and Lönnqvist built two prototype systems, fuseONE and fuseTWO, which used their WISP (weakly intrusive ambient soundscape) [74] to communicate states and events in computational and physical environments through auditory cues.

RAVE

The Ravenscroft Audio Video Environment (RAVE) is described by Gaver et al. [50]. The motivation behind RAVE was to support collaborative working amongst colleagues dispersed throughout several rooms in a building. Each room had an audio-video node which comprised a camera, monitor, microphone, and speakers. All items in the node could be moved and turned on and off at will by the users. Users interacted with RAVE through GUI buttons. For example, pressing the background button would select a view from one of the public areas to be displayed on the audio-video node's monitor screen. The sweep button would cause a short sampling of each node's camera allowing users to find out who is present in the various offices. A specific location could be viewed by a three-second glance to its camera. Offices could also be connected via the *vphone* and *office share* functions which allowed creation of

a larger virtual office. Auditory icons were introduced to allow greater levels of reasonable privacy. For example, when a glance connection was requested, an auditory warning was sounded at the target node's location three seconds prior to the glance being activated. When connections were broken other sounds, such as that of a door closing, were triggered. These auditory icons gave information about system state and user interactions.

OutToLunch

Cohen took the idea of monitoring background activity further with his OutToLunch system [33]. Cohen states:

> The liquid sound of keystrokes and mouse clicks generated by a number of computer users gives co-workers a sense of 'group awareness' — a feeling that other people are nearby and an impression of how busy they are. When my group moved from a building with open cubicles to one with closed offices, we could no longer hear this ambient sound. [p. 15]

OutToLunch was an attempt to restore this sense of group awareness through the use of auditory icons and an electronic sign board. Each time a user hit a key or clicked a mouse, a corresponding click sound would be played to the other users. Total activity over a thirty second period, rather than the exact timing of individual key strokes and mouse clicks, was recorded. Because the sounds did not convey much information most users soon found the system annoying. Unlike a real shared workspace, OutToLunch did not give directional clues to allow users to determine who was making the key clicks or mouse clicks. In a revised system each person in the group was represented by their own unique musical motif.

Workspace Zero

Eggen et al. [40] wanted to explore the interactions between people and their environment with a focus on understanding how a pervasive soundscape affects the behavior of an environment's inhabitants. With careful sound design they discovered that not only can sound be useful as an information carrier but it also has a "decorative value". Eggen et al. were greatly influenced by the concept of "calm technology" [127] which, in the context of auditory display, allows the user of a sonification to easily switch the sound between the perceptual foreground and background.

18.4 Monitoring program execution

Talk to engineers from the 1950s, '60s and '70s (and even home computer enthusiasts from the '80s) and many will tell stories about how they tuned AM radios to pick up the electrical noise given out by the processors of their computers. They learned to recognize the different patterns of sounds and matched them to program behavior during debugging activities [122] (see also section 9.12 in this volume). In fact, the use of audio as a tool for monitoring and debugging programs would seem to have a history almost as long as computer science itself. In his 2011 BCS/IET Turing Lecture Donald Knuth [76] made reference to the Manchester Mark 1 computer of 1949 (the successor to Baby, the world's first stored program computer) which had its circuits wired to an audio channel so that programs could be debugged by listening to them. Despite four decades of radio-assisted debugging it was not until the 1990s that research articles exploring the issue systematically began to appear. The case for using sound to aid programming was supported by Jackson and Francioni [64], although they felt that a visual resentation was also needed to provide a context or framework for the audio sound track. They argued that some types of programming error (such as those that can be

spotted through pattern recognition) are more intuitively obvious to our ears than our eyes. Also, they pointed out that, unlike images, sound can be processed by the brain passively, that is, we can be aware of sounds without needing to listen to them. Francioni and Rover [49] found that sound allows a user to detect patterns of program behavior and also to detect anomalies with respect to expected patterns.

One of the first attempts at program sonification was described by Sonnenwald et al. [105] and was followed by DiGiano [37], DiGiano and Baecker [38], Brown and Hershberger [24], Jameson [68, 67], Bock [12, 13, 14], Mathur et al. [82, 10] and Vickers and Alty [121, 119, 120, 122, 118, 123]. These early systems all used complex tones in their auditory mappings but, like much other auditory display work, this was done without regard to the musicality of the representations (with the exception of Vickers and Alty). That is, simple mappings were often employed, such as quantizing the value of a data item to a chromatic pitch in the 128-tone range offered by MIDI-compatible tone generators. Furthermore, the pitches were typically atonal in their organization and were combined with sound effects (e.g., a machine sound to represent a function processing some data). Effort was largely invested in demonstrating that data could be mapped to sound with much less attention given to the aesthetics and usability of the auditory displays. Where aesthetic considerations are taken into account auditory displays become much easier to comprehend. Mayer-Kress, Bargar, and Choi [83] mapped chaotic attractor functions to musical structures in which the functions' similar but never-the-same regions could be clearly heard. The resultant music could be appreciated in its own right without needing to know how it was produced. Quinn's Seismic Sonata [90] likewise uses the aesthetics of musical form to sonify data from the 1994 Northridge, California earthquake. Alty [1] demonstrated how musical forms could be used to sonify the bubble-sort algorithm. When we turn from pure data sonification and look towards sonification techniques as a complement to existing visualization models we find that good progress has been made. Brown and Hershberger [24] coupled visual displays of a program during execution with a form of sonification. They suggested that sound will be a "powerful technique for communicating information about algorithms".

Brown and Hershberger offered some examples of successful uses of audio, for instance, applying sound to the bubble-sort algorithm. The main use of sound in this work was to reinforce visual displays, convey patterns and signal error conditions and was by no means the main focus of the work. Like most other visualization systems that employ audio, Brown and Hershberger's work used sound as a complement to visual representations. No formal evaluation of the approach was published. Early efforts at pure sonification were concerned with specific algorithms, often in the parallel programming domain. Examples include Francioni et al. [47, 48] who were interested in using sonifications to help debug distributed-memory parallel programs, and Jackson and Francioni [65, 64] who suggested features of parallel programs that would map well to sound. Those systems that are applied to more general sequential programming problems require a degree of expert knowledge to use, whether in terms of programming skill, musical knowledge, expertise in the use of sound generating hardware, or all three.

18.4.1 Systems for external auditory representations of programs

To date the following program sonification tools have been identified (as opposed to visualization systems that incorporate some sonification): InfoSound [105] by Sonnenwald

et al, the LogoMedia system by DiGiano and Baecker [38], Jameson's Sonnet system [68, 67], Bock's Auditory Domain Specification Language (ADSL) [12, 14, 13] the LISTEN Specification Language, or LSL [82, 11, 10], Vickers and Alty's CAITLIN system [121, 119, 120, 122, 118, 123], Finlayson and Mellish's AudioView [41], Stefik et al. [109] and Berman and Gallagher [8, 7].[8]

Infosound Sonnenwald et al.'s InfoSound [105] is an audio-interface tool kit that allows application developers to design and develop audio interfaces. It provides the facility to design musical sequences and everyday sounds, to store designed sounds, and to associate sounds with application events. InfoSound combines musical sequences with sound effects. A limitation is that the software developer is expected to compose the musical sequences himself. To the musically untrained (the majority) this militates against its general use. The system is used by application programs to indicate when an event has occurred during execution and was successfully used to locate errors in a program that was previously deemed to be correct. Users of the system were able to detect rapid, multiple event sequences that are hard to detect visually using text and graphics.

LogoMedia DiGiano and Baecker's LogoMedia [38], an extension to LogoMotion [2] allows audio to be associated with program events. The programmer annotates the code with probes to track control and data flow. As execution of the program causes variables and machine state to change over time, the changes can be mapped to sounds to allow execution to be listened to. Like InfoSound, LogoMedia employs both music and sound effects. DiGiano and Baecker assert that comprehending "the course of execution of a program and how its data changes is essential to understanding why a program does or does not work. Auralisation expands the possible approaches to elucidating program behavior" [38].[9] The main limitation is that the sonifications have to be defined by the programmer for each expression that is required to be monitored during execution. In other words, after entering an expression, the programmer is prompted as to the desired mapping for that expression.

Sonnet Jameson's Sonnet system [68, 67] is specifically aimed at the debugging process. Using the Sonnet visual programming language (SVPL) the code to be debugged is tagged with sonification agents that define how specific sections of code will sound. The sound example is of a bubble sort algorithm with an indexing error bug (sound example **S18.3**). Figure 18.1 shows a component to turn a note on and off connected to some source code. The "P" button on the component allows static properties such as pitch and amplitude to be altered. The component has two connections, one to turn on a note (via a MIDI note-on instruction) and one to silence the note (MIDI note-off). In this example the note-on connector is attached to the program just before the loop, and the note-off connector just after the close of the loop. Therefore, this sonification will cause the note to sound continuously for the duration of the loop. Thus, placement of the connectors defines the sonification.

[8] Stefik and Gellenbeck [108] have latterly reported their Sonified Omniscient Debugger in which speech is used as the auditory display.

[9] *Auralisation* was the term originally adopted by researchers working in the program sonification domain. It has fallen out of common usage with authors tending to use the more general (and well-known) *sonification*, so auralisation is now a (mostly) deprecated term.

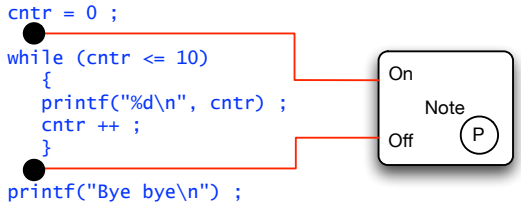


Figure 18.1: A Sonnet VPL Component redrawn version of the original in Jameson [68]. The VPL component is the box on the right.

Other components allow the user to specify how many iterations of a loop to play. Program data could also be monitored by components that could be attached to identifiers within the code. The guiding principle is that the programmer, knowing what sounds have been associated with what parts of the program, can listen to the execution looking out for patterns that deviate from the expected path. When a deviation is heard, and assuming the expectation is not in error, the point at which the deviation occurred will indicate where in the program code to look for a bug.

Sonnet is an audio-enhanced debugger. This means that because it interfaces directly with the executing program it is not invasive and does not need to carry out any pre-processing to produce the sonifications. The visual programming language offers great flexibility to the programmer who wants to sonify his program. However, it does require a lot of work if an entire program is to be sonified even very simply.

```
Track_name=Loop
{
1 Track=Status('for'):Snd("for_sound");
2 Track=Status('while'):Snd("while_sound");
}
```

Figure 18.2: An ADSL track to monitor for and while loops redrawn from the original from Bock [12]

ADSL

Bock's Auditory Domain Specification Language (ADSL) [12, 14, 13] differs from the above three approaches in that it does not require sounds to be associated with specific lines of program code. Instead users define tracks using the ADSL meta-language to associate audio cues with program constructs and data. These tracks (see Figure 18.2) are then interpreted by a pre-processor so that the code has the sonifications added to it at compilation allowing the program to be listened to as it runs. The fragment of ADSL in Figure 18.2 specifies that `for` and `while` loops are to be signalled by playing the '`for_sound` and '`while_sound`' respectively. These two sounds have been previously defined and could be a MIDI note sequence, an auditory icon, or recorded speech. The sounds will be heard when the keywords `for` and `while` are encountered in the program. An advantage of this approach is that it is possible to define a general purpose sonification. That is, by specifying types of program construct to be sonified there is no requirement to tag individual lines of code with sonification specifications. When such tagging is required this can be done using the features of the

```
begin auralspec
specmodule call_auralize
var
  gear_change_pattern, oil_ check_ pattern, battery_ weak_pattern: pattern;
begin call_auralize
  gear_ change_pattern:="F2G2F2G2F2G2C1:qq'"+ "C1:";
  oil_ check_pattern:="F6G6:h"';
  battery_weak_pattern:="A2C2A2C2";
  notify all rule = function_ call "gear_change" using gear_ change_ pattern;
  notify all rule = function_ call "oil_ check" using oil_ check_ pattern;
  notify all rule = function_ call "battery_ weak" using battery_ weak_ pattern;
end call_auralize;
end auralspec.
```

Figure 18.3: An LSL ASPEC for an automobile controller redrawn version of the original in Mathur et al. [82]

specification language. Like Sonnet, ADSL is non-invasive as the sonifications are added to a copy of the source program during a pre-processing phase.

ADSL uses a mixture of digitized recordings, synthesized speech, and MIDI messages. The choice of sounds and mappings is at the discretion of the user, although a common reusable set of sonifications could be created and shared amongst several programmers. Tracks could be refined to allow probing of specific data items or selective sonification of loop iterations. The system is flexible, allowing reasonably straightforward whole-program sonification, or more refined probing (such as in Sonnet).

In a study [13] thirty post-graduate engineering students from Syracuse University all with some programming experience were required to locate a variety of bugs in three programs using only a pseudo-code representation of the program and the ADSL auditory output. On average, students identified 68% of the bugs in the three programs. However, no control group was used, so it is not possible to determine whether the sonifications assisted in the bug identification.

Listen/LSL & JListen

Mathur's Listen project [82, 10] (currently inactive) follows a similar approach to that used by ADSL. A program is sonified by writing an "auralisation specification" (ASPEC) in the Listen Specification Language (LSL). A pre-processing phase is used to parse and amend a copy of the source program prior to compilation. Again, the original source program is left unchanged. The accompanying sound file is LISTEN's sonification of a bubble sort algorithm (sound example **S18.4**). An ASPEC defines the mapping between program-domain events and auditory events, and an example for an automobile controller is shown in Figure 18.3. This example sonifies all calls to the program functions `gear_change`, `oil_check` and `weak_battery`. The ASPEC contains the sonification definitions and their usage instructions. For instance, in Figure 18.3 we see that a call to program function `gear_change` causes the sonification `gear_change_pattern` to be played. This sonification is defined earlier in the ASPEC as a sequence of MIDI note-on/off instructions.

As LSL is a meta-language it can, in theory, be used to define sonifications for programs written in any language; the practice requires an extended version of LSL for each target language. What is immediately apparent is that writing ASPECS in LSL is not a trivial task as it requires the programmer to learn the syntax of LSL. In addition, some musical knowledge is needed to know how to specify which pitches are used in the sonifications. The Listen project appeared dormant after 1996 but was reactivated in 2002 and applied to Java

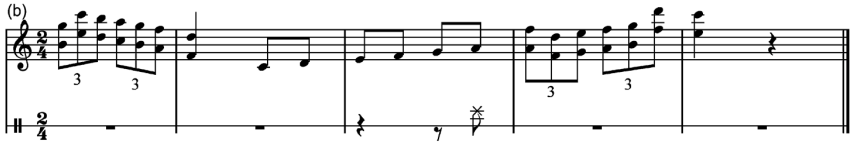
in the form of JListen [30]. Some informal studies and applications of JListen have been carried out (see Gopinath [56] and Prasath [89]) but, to date no formal experimentation or evaluation of the original or newer JListen systems has been published. The CAITLIN system

(a)

```
FOR counter := 1 TO 6 DO
   counter := counter + 1 ;
```

The Pascal code (a) results in the auralization (b). Bar 1 and beat 1 of bar 2 is the tune denoting entry to the loop. The notes remainder of bar 2 and bar 3 representsix iterations of the loop, the final iteration being supplemented by a sleighbell sound. Bars 4 and 5 denote exit from the loop.

(b)



(c)

```
a : = 10 ;
CASE a OF
   '1' : Writeln ('Found 1') ;
   '2' : Writeln ('Found 2') ;
   '3' : Writeln ('Found 3') ;
   '4' : Writeln ('Found 4') ;
   ELSE  Writeln ('Not found)
END ;
```

The Pascal code (c) results in the auralization (d). Bar 1 and beat 1 of bar 2 is the tune denoting entry to the CASE construct. A cowbell sound is played as each case instance is tested (bars 2, 3, and 4). The tune in bars4 to 6 signify exit from the construct. In this example, no match was found for the case selector, so the construct exits in a minor key. In the CAITLIN system the major mode was used to denote Boolean true and minor for Boolean false.

(d)

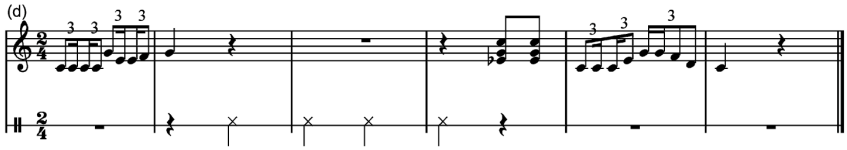


Figure 18.4: Example CAITLIN sonifications taken from Vickers [115]

CAITLIN

is a non-invasive pre-processor that allows the sonification of programs written in Turbo Pascal [121, 119, 120, 122, 118, 123]. Musical output is achieved by sending MIDI data to a multi-timbral synthesizer via the MIDI port on a sound card. The CAITLIN sonifications were designed around a tonal musical framework. Users were not required to design the sonification content as the system used unique pre-defined motifs (theme tunes) to represent the Pascal language constructs [120]. This meant that no musical knowledge was needed in order to produce musically-consistent output. Figure 18.4 shows example sonifications for two program fragments. Figure 18.4(d) can be heard in the accompanying sound file (sound example **S18.5**). Experiments with the CAITLIN system indicated that musical sonifications do provide information useful in bug-location and detection tasks [122, 123]. As the CAITLIN system was an experimental prototype it has some limitations. First, the sonifications were only applied to the language constructs (selections and iterations) which meant that other program features could not be inspected aurally. Secondly, constructs could not be individually marked for sonification meaning the entire program was sonified. In a real debugging situation programmers would not monitor an entire program but would choose candidate sections for close scrutiny. This whole-program approach meant that even short programs could take a long time to play back.

AudioView

Finlayson and Mellish [41] developed the AudioView to provide an auditory *glance* or overview of Java source code. Using non-speech-audio-only, speech-only, and combined non-speech & speech output modes, Finlayson and Mellish explored how good the auditory channel is at helping programmers to gain higher level understanding of their programs. A similar approach to Vickers and Alty was taken in the construction of the non-speech cues in that a structured hierarchy of earcons was built which corresponded to the hierarchy of Java programming constructs (see Figure 18.5).[10] Finlayson, Mellish, and Masthoff [42] extended the AudioView system to provide auditory fisheye views of the source code.

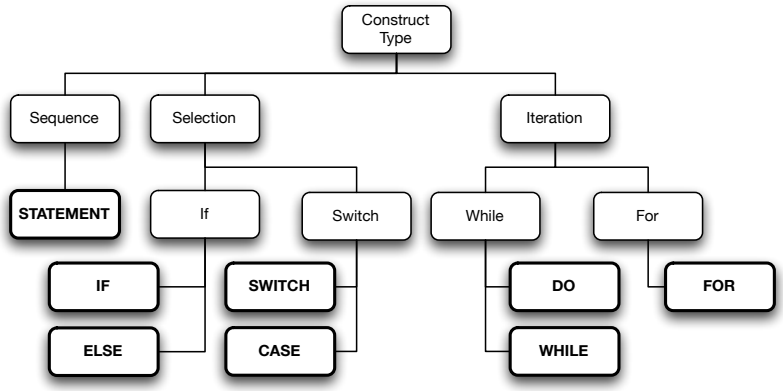


Figure 18.5: Basic Java construct hierarchy redrawn version of original from Finlayson and Mellish [41]

Program slices and low-level structures

Berman and Gallagher [7, 8] were also interested in higher-level or aggregated structural knowledge. To assist with debugging, rather than sonify the entire run-time behavior of a program they computed *slices* (sets of statements that are implicated in the values of some user-selected point of interest), (sound example **S18.6**). In a series of small studies Berman and Gallagher [7] found that programmers were able to use sonifications of the slices to infer knowledge such as the homgeneity of the slice, the amount of a particular function/method participating in a slice, etc. In follow-up work [8] they combined CSound (a music synthesis and signal processing system) with the Eclipse integrated development environment (IDE) to provide a framework for sonifying low-level software architecture.[11] The tool allowed programmers to sonify information about low-level and static program structures such as packages, classes, methods, and interfaces.

Sonification for program monitoring and debugging received the majority of its interest in the 1990s and the first decade of this century and examples of program sonification have been less common in recent years. This is not to say that software visualization has diminished *per se* (for example, see Romero et al. [96, 95, 94, 93]) but sonification work appears to have been focused more in other areas of endeavor. Perhaps this is related to the relative complexity of programming environments. In the 1980s a good deal of programming was done with

[10] Vickers and Alty did such a hierarchical analysis for the Pascal language [120].

[11] For information on CSound see http://www.csounds.com/. For Eclipse, see http://www.eclipse.org/.

line editors and command-line compilers on dumb mainframe terminals and the procedural programming paradigm was dominant. The late 1980s witnessed the start of the large-scale take up of the IBM PC-compatible machines and integrated development environments (IDE) began to get a foothold. Despite the more powerful programming environments graphical representations were still primitive and sonification was directed at communicating the run-time behavior of programs. Today, object-oriented programming is the prevailing paradigm, IDEs are much richer, code is concurrent and multithreaded, and programmers are looking for tools that offer insight into much more than run-time behavior. There are exceptions, of course. For example, Lapidot and Hazzan [78] describe a project that sets out to use music as a framework for helping computer science students to reflect upon the activity of debugging. Stefik and Gellenbeck [108] explored the use of speech for assisting the debugging process; although this is not, strictly, sonification, it is encouraging that the problem still continues to receive attention. What is particuarly interesting in Stefik and Gellenbeck's research is the reason for using speech rather than sonification techniques:

> While we find the idea of musical structures to represent code structures interesting, we have tested hundreds of combinations of musical structures in our laboratory and we generally find them unlikely to produce comprehension benefits in general-purpose integrated development environments. The reason for this may be that an obvious, intuitive, mapping between arbitrary program constructs and music does not exist. [110, p. 70]

Moving beyond the problem of run-time behavior Berman and Gallagher [7] and Finlayson et al. [41, 42] have provided insight into what future program monitoring environments will need to include for sonification to begin to get wider acceptance. Given the increased potential for external auditory representations of software it is somewhat suprising that more effort has not been engaged in such research. Hussein et al. [63] suggest that the main obstacle is that multi-disciplinary teams are needed with expertise in both music and software technology. However, this is arguably true for all sonification research (see Chapter 7) and so it is likely that the reasons are varied. Stefik and Gellenbeck [108] observe that understanding and debugging programs are intrinsically difficult activities and this might be one factor in the relative paucity of program sonification research. Another might be related to the IDEs that are used today. In the early days of program sonification work there were few IDEs and graphical programming environments were often the result of specialist research projects. Today there are several popular IDEs in use (e.g., Eclipse and NetBeans) and for sonification to make any real inroads it needs to be integrated with these environments. In the past researchers tended to build bespoke sonification environments, but this is no longer necessary. It is encouraging that the more recent projects have adopted the integrating-with-existing-IDEs approach and this would seem to be the way forward.

18.5 Monitoring interface tasks

SonicFinder

Gaver was one of the first to propose adding sounds to a general computer interface to allow users to monitor the progress and state of sundry activities and tasks. His SonicFinder system [53] used auditory icons [51] to enhance the existing Apple Macintosh Finder program. The visual feedback provided by the Finder was extended by adding sounds. The auditory icon for a progress bar representing a file copy operation was the sound of a jug being filled with

water. An empty jug sounds hollow and the water pouring into it has a low pitch. As the file copy progressed, so the pitch of the water sound rose and the jug sounded increasingly full. The auditory icon did not tell the user anything that the visual display did not communicate, serving only to reinforce the feedback. However, unlike the visual version which showed the overall length of the progress bar and the amount completed at any point, the auditory icon gave no reference points for the start and end of the task. Thus, whilst the listener could hear the progress of the file copy, and could possibly infer that it was nearing its end, no absolute information to this effect was provided aurally. Gaver reports that the SonicFinder was appealing to users and that some were reluctant to give it up.[12] Unfortunately, no formal empirical or other evaluation was carried out so there is no firm evidence to indicate the efficacy of Gaver's system.

Lumsden et al. [80] identified three important principles for successful interface sonification widgets:

1. Minimize the annoyance
2. Simplify the mapping
3. Facilitate segregation of the sonification elements

Brewster has worked extensively with earcons to sonify interface activities [18, 17, 22, 19, 21, 23, 20]. Of particular interest here is Crease and Brewster's investigation of auditory progress bars [35, 36]. Parallel earcons were used to represent the initiation, progress, heartbeat, remainder, and completion information of a progress bar activity (these terms come from Conn's task properties for good affordance [34]). Experimentation with participants showed that users preferred the auditory progress bar to a visual one, that frustration experienced by participants fell in the auditory condition, and that annoyance levels were roughly the same for the auditory and visual progress bars. Furthermore, Crease and Brewster commented that the addition of sounds "allows users to monitor the state of the progress bar without using their visual focus ... participants were aware of the state of the progress bar without having to remove the visual focus from their foreground task" [36].

18.5.1 Web server and internet sonification

Two good candidates for live monitoring applications are web servers and computer networks.

Peep

Gilfix and Crouch [55] said of activity in complex networks that it is "both too important to ignore and too tedious too watch" [p. 109]. Their solution to this conundrum was to build the Peep network sonification tool. Peep employed a soundscape approach (an "'ecology' of natural sounds" in their words) in which network state information was gathered from multiple sources and was used to trigger the playback of pre-recorded natural sounds (sound example **S18.7**). This mapping between network events and natural sounds created a soundscape that represented the continuous and changing state of the network being monitored. Like most sonifications the mappings did not allow the communication of absolute values by the sounds but the person monitoring the network was able to infer rich knowledge about the network's state simply through hearing the relative differences

[12] One could also argue that the novelty factor of an auditory interface in 1989 would have added to the SonicFinder's appeal, but interface event sounds are now commonplace in modern desktop operating systems and so the novelty has worn off. However, it is interesting that despite Gaver's work back in 1989, auditory progress bars themselves still do not come as standard in operating system interfaces.

between and changes in the individual components of the soundscape. The power of this representational technique lay in the human auditory system's ability to recognise changes in continuous background sounds.

Ballora et al. [4] undertook a similar exercise to Gilfix and Crouch but instead of using pre-recorded natural sounds they used a technique more akin to model-based sonification in which the various channels of network traffic data being monitored drove the values of the parameters of a sound model in the SuperCollider synthesis program.[13] In their case they experimented with several approaches in which the the four octet values of an IP address controlled:

- The levels of the four main partials in a vibraphone model;
- The parameters of four separate instances of a water-like instrument;
- A set of formants in a vocal model.

They found that the mappings resulted in sonic backdrops that were neither annoying nor distracting.

WebMelody

Barra et al. [6, 5] sonified web server workload, severe server errors, and normal server behavior, by mapping the server data to musical soundtracks with their WebMelody system. Echoing Tran and Mynatt (see section 18.3.3) the motivation behind WebMelody was to allow long-term monitoring of the web server with minimum fatigue. A principal benefit of the auditory approach is that it is eyes-free. In an attempt to minimize fatigue and annoyance, Barra et al. moved beyond the use of simple audio alarms and tried "to offer an aesthetic connotation to sounds that border between music and background noises." [5, p.34]. In so doing it was claimed that WebMelody "let users listen for a long time without diverting them from their work". The approach taken was to allow the user to select any external music source of his or her choosing into which MIDI-based sonifications of the web server data were mixed. This allowed the administrator to hear the status of the server while listening to his or her preferred choice of music (sound example **S18.8**). One claimed benefit of WebMelody is that it allows real-time feedback to be provided, something that general log analyzers cannot do given their reliance on aggregated log files [6]. WebMelody uses an information push approach rather than the more typical pull needed to get information from a server's log file.[14] A potential drawback of real-time auditory information push is that the auditory stream could become tiring and/or annoying, as well as being environmentally intrusive (the stream is only of interest to those wishing to monitor the server activity). To mitigate these effects, WebMelody's designers made the system as configurable as possible.

Table 18.1 summarizes the characteristics of some of the systems discussed above showing their domain of application, their type (direct, peripheral, serendipitous-peripheral) and the primary sonification technique employed.

[13] SuperCollider is available from http://supercollider.sourceforge.net/.

[14] This is analogous in concept to using a data stream connection (WebMelody) instead of a state vector connection (log file analysis) in Jackson's JSD nomenclature [66].

System	Domain	Type (D,P,S-P)[1]	Sonification technique employed
Audio Aura (Mynatt et al)	Workplace monitoring	S-P	Sonic landscapes
OutToLunch (Cohen [33])	Workplace activities	P & S-P	Sound effects/auditory icons
ShareMon (Cohen [32])	Monitoring of file sharing	P & S-P	Sound effects/auditory icons
fuseONE, fuseTWO (Kilander and Lönnqvist [75])	Remote process monitoring	P & S-P	Ambient soundscapes
Music Monitor (Tran & Mynatt [113])	Monitoring domestic environments	S-P	Tonal music & ambient soundscapes
ListenIN (Schmandt & Vallejo [101])	Monitoring domestic environments	P & S-P	Auditory icons
LogoMedia (DiGiano & Baecker)	Program monitoring	D & P	Ad-hoc pitch mappings & sound effects
LISTEN/LSL (Mathur et al)	Program monitoring & debugging	D & P	Ad-hoc pitch mappings
ADSL (Bock [12, 14, 13])	Program monitoring & debugging	D & P	Ad-hoc pitch mappings
Sonnet (Jameson [68, 67])	Program monitoring & debugging	D & P	Ad-hoc pitch mappings
Zeus (Brown & Hershberger [24])	Program monitoring	D & P	Simple pitch mappings
Parallel programs (Jackson & Francioni [64])	Monitoring of parallel programs	D & P	Simple pitch mappings
Infosound (Sonnenwald et al. [105])	Monitoring of parallel programs	D& P	Ad-hoc pitch mappings
Nomadic Radio (Sawhney [98])	Contextual messaging in roaming environments	D & P	Speech & non-speech audio
SonicFinder (Gaver [53])	Interface monitoring	D & P	Auditory icons
ARKola (Gaver, Smith, & O'Shea [54])	Industrial plant monitoring	D	Auditory icons
CNC Robots (Rauterberg & Styger [91])	Industrial plant monitoring	D	Auditory icons
AudioView (Finlayson & Mellish [41])	Program monitoring & debugging	D & P	Earcons
Berman and Gallagher [8, 7]	Program architecture	D & P	Musical motifs
CAITLIN (Vickers & Alty [121, 119, 120, 122, 118, 123])	Program monitoring & debugging	D & P	Tonal music motifs/earcons

[1] (D)irect, (P)eripheral, (S-P)Serendipitous-peripheral

Table 18.1: Summary characteristics of a selection of monitoring systems.

18.6 Potential pitfalls

Previous sections showed how researchers have attempted to provide auditory displays for direct, peripheral, and serendipitous-peripheral auditory process monitoring. From this body of work we may identify a number of principal challenges that face designers of such sonifications:

1. The potential intrusion and distraction of sonifications;
2. Fatigue & annoyance induced by process sonification;
3. Aesthetic issues and acoustic ecology;
4. Comprehensibility and audibility.

18.6.1 Intrusion and distraction, fatigue and annoyance

There is a tension when designing auditory process monitors between the sonification being perceptible to its intended audience and being too intrusive or annoying. In their work on awareness support systems, Hudson and Smith [62] commented on the problem of intrusion in terms of awareness and privacy. They stated that this "dual tradeoff is between privacy and awareness, and between awareness and disturbance". The more information an auditory monitor provides the richer the sonification yet the greater the potential for disturbance, annoyance, and an upset in the balance of the acoustic ecology. Gutwin and Greenberg [57] claimed it is a tradeoff between being well informed and being distracted. Kilander and Lönnqvist [75] noted the effect of such sonifications on people sharing the workspace who are not part of the monitoring task:

> In a shared environment, one recipient may listen with interest while others find themselves exposed to an incomprehensible noise. [p. 4]

Indeed, commenting upon the design of their *nomadic radio* system, Sawhney and Schmandt [97, 98] cautioned that care must be taken to ensure that the auditory monitoring system intrudes minimally on the user's social and physical environment.[15]

In dealing with intrusiveness, Pedersen and Sokoler [88] framed the problem as a balance between putting a low demand on attention versus conveying enough information. In their Aroma system the goal was to communicate knowledge and awareness of the activities of people at remote sites, thus the privacy of the people being monitored by the system was of great importance. They studied this problem through an "ecology of awareness" showing awareness of the importance of the acoustic ecology of a sonification. Pedersen and Sokola made the auditory, visual, and haptic representations of the Aroma system highly abstract — abstraction would allow useful information to be communicated without divulging too many details that would violate privacy. It was hoped that abstract representations would be better at providing "peripheral non-attention demanding awareness" [88, p. 53]. It was also noted that such abstract representations lend themselves to being remapped to other media (what Somers [104] would call *semiotic transformation*), or, in turn, foster accommodating user preferences (an important aspect of aesthetic computing). Unfortunately, user studies showed that the

[15] In *nomadic radio* a mixture of ambient sound, recorded voice cues, and summaries of email and text messages is used to help mobile workers keep track of information and communication services.

abstraction led to users interpreting the representations in varied ways [87]. Furthermore, Kilander and Lönnqvist [75] warned that the "monitoring of mechanical activities such as network or server performance easily runs the risk of being monotonous" and Pedersen and Sokola reported that they soon grew tired of the highly abstract representations used in Aroma. It is interesting that they put some of the blame down to an impoverished aesthetic, feeling that involving expertise from the appropriate artistic communities would improve this aspect of their work.

Cohen [32] identified a general objection to using audio for process monitoring: people in shared office environments do not want more noise to distract them. Buxton [26] argued that audio is ubiquitous and would be less annoying if people had more control over it in their environments. Lessons from acoustic ecology would be helpful here. Cohen [32] defined an acoustic ecology as "a seamless and information-rich, yet unobtrusive, audio environment".

Kilander and Lönnqvist [75] tackled this problem in their fuseONE and fuseTWO environments with the notion of a *weakly intrusive ambient soundscape* (WISP). In this approach the sound cues for environmental and process data are subtle and minimally-intrusive.[16] Minimal- or weak-intrusion is achieved in Kilander and Lönnqvist's scheme by drawing upon the listener's expectation, anticipation, and perception; anticipated sounds, say Kilander and Lönnqvist, slip from our attention. For example, a ticking clock would be readily perceived and attended to when its sound is introduced into the environment (assuming it is not masked by another sound). However, as the steady-state of the ticking continues and the listener expects or anticipates its presence its perceived importance drops and the sound fades from our attention [75]). However, a change in the speed, timbre, or intensity of the clock tick would quickly bring it back to the attention of the listener. Intrusiveness can thus be kept to a necessary minimum by using and modulating sounds that fit well with the acoustic ecology of the process monitor's environment. The sonification is then able to be discriminated from other environmental sounds (either by deliberate attentiveness on the part of the listener, or by system changes to the sounds), yet is sufficiently subtle so as not to distract from other tasks that the listener (and others in the environment) may be carrying out. To increase the quality of the acoustic ecology further, Kilander and Lönnqvist used real-world sounds rather than synthesized noises and musical tones. They concluded that

> ...easily recognisable and natural sounds ...*[stand]* ...the greatest chance of being accepted as a part of the environment. In particular, a continuous background murmur is probably more easily ignored than a singular sound, and it also continuously reassures the listener that it is operative. [75]

Kilander and Lönnqvist's weakly intrusive ambient soundscapes would thus seem to be a suitable framework for the design of peripheral process monitoring sonifications in which monitoring is not the user's primary or sole task.

Schmandt and Vallejo [101] noted the perception-distraction dichotomy. Their ListenIN system for monitoring activity in a domestic environment attempted to provide continuous but minimally-distracting awareness. Unfortunately, no formal studies have been carried out with the system to test this aim. Mynatt et al. [85] aimed with their Audio Aura

[16] Kilander and Lönnqvist actually used the adjective 'non-intrusive' to describe their sonifications. One could argue that this term is misleading as any sonification needs to be intrusive to some extent in order to be heard. Their term 'weakly intrusive' is more helpful and more accurate.

scheme to provide environmental sonifications that enriched the physical world without being distracting. Tran and Myatt reduced the intrusiveness of their Music Monitor system [113] by overlaying earcons on top of music tracks chosen by the main user. The mixing of earcons with intentional music meant that other people in the environment would not be distracted as the encoded messages in the earcons would only be recognised by the main user: other people would just be aware of changes in the music. Of course, the fact that there is a music stream at all means that the system does still intrude into the environment. Rather, the attempt here was to minimize the negative distracting effects of that intrusion. Preliminary experimental results showed that the music was not distracting to those who were not monitoring the earcon messages embedded within it [113]. In a related project that combined auditory displays with tangible computing Bovermann et al. [16, 15] found that their Reim toolset could be used to provide peripheral monitoring without causing distraction or annoyance.

Fatigue is sometimes mentioned as a potential problem associated with auditory display but it is notable that hearing is "more resistant to fatigue than vision" [91, p. 42] and so it is not clear that auditory displays should cause more problems in this regard than visual representations.

18.6.2 Emotive associations

The degree and detail to which process data are sonified depends a great deal on the intended audience. Some may take a dispassionate view whilst others may attach emotional significance to the data. Cohen [31] found that it is difficult to construct "sounds which tell the right story and are also pleasant and emotionally neutral". For example, in their work on sonifying weather reports Hermann, Drees, and Ritter [59] noted that whilst meteorologists would be interested in exploring and analyzing long-term time-series weather data, the average public consumer of a weather forecast requires a much more abstract view in terms of what will the weather be this afternoon, tomorrow, or at the weekend. Choice of activity and clothing are dependent on the weather, so a simple forecast indicating likely temperature, wind, and precipitation levels is sufficient for most tasks. Furthermore, a weather forecast can trigger an emotional response in the listener. Hermann, Drees, and Ritter put it that rather than having the detailed quantitative interest of the expert meteorologist, the "listener is concerned with *[the weather's]* emotional value and contextual implications, which are not simply assessed from single ... attributes like temperature or humidity in isolation" [59]. For instance, to a northern European a temperature of 30^{o}C would be very pleasant if the humidity is low, but quite unpleasant in overcast humid conditions. Thus, the sonification designer can take into account the fact that whilst the raw process data themselves are free of emotive content, their values and combinations can cause an inferential process of emotional coding on the part of the listener. Hermann, Drees, and Ritter [59] attempted to deal with this issue directly by deriving emotional relations from the high-dimensional 'weather vector'. This was accomplished by constructing an *Emo-Map* (see Figure 18.6). The Emo-Map is a two-dimensional plot of hourly weather vectors: each vector is displayed as a single point on the plot. From this plot were derived a number of prototype weather states (such as 'hot dry summer day', 'snowy winter day' and 'golden October day'). Each prototype had an associated emotive aspect (e.g., enervated and indifferent for the hot dry summer day, negative, calm, and indifferent for the snowy winter day, and positive emotional state for the

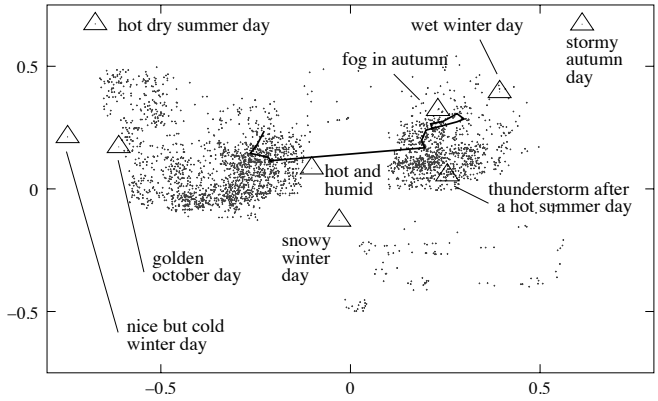


Figure 18.6: The 'Emo-Map' from Hermann et al. [59] showing some typical weather states and prototype markers.

golden October day). Sounds were chosen to correspond to the prototype-emotion classes (panting sound & cricket songs for the summer day, a shudder/shiver sound for the snowy winter day, and a rising 'organic sound' for the golden October day). A resultant sonification is shown schematically in Figure 18.7.

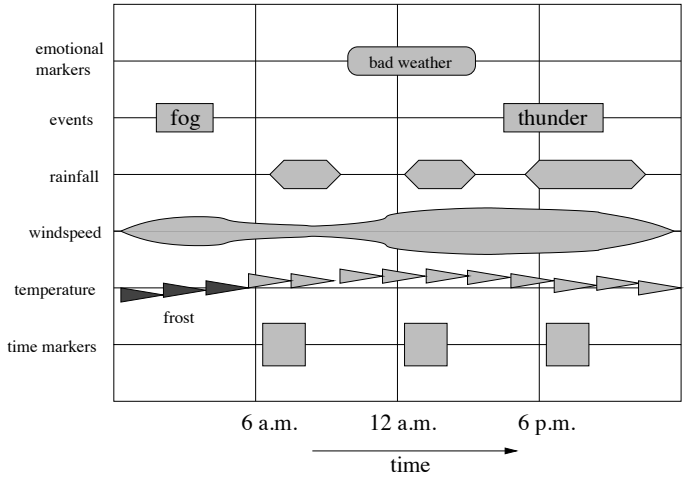


Figure 18.7: Graphical visualization of Hermann et al.'s weather sonification [59]. The y axis shows the six auditory streams with time shown along the x axis. As humidity and wind direction were not represented by their own streams but modulated existing streams, they are not shown on the figure.

18.6.3 Aesthetics and acoustic ecology

In some of the work discussed above, authors used terms such as *ecology* and *acoustic ecology* in reference to their sonification designs. In recent years there has been a growing realization of the important role to be played by aesthetics in the design of computer systems and artefacts. Fishwick [43, 44, 45] coined the term *aesthetic computing* to refer to application of art theory and practice to the design of computing systems. Fishwick [44] claims

> ...there is a tendency toward the mass-media approach of standardized design, rather than an approach toward a more cultural, personal, and customized set of aesthetics.

It is making use of cultural and personal differences that Fishwick claims will enlarge "the set of people who can use and understand computing". Whilst Fishwick has focused primarily on the aesthetics of visualizations and models, recent research in the auditory display community has begun to pay attention to aesthetic issues in sonification.[17]

Cohen used acoustic ecologies of sounds in his ShareMon system, only he called them collections of *genre* sounds [31]. A principle of aesthetic computing is that systems should be malleable according to the culture in which it is situated [43]. Cohen argued the strongest reason for *not* using 'genre' sounds is that they are less universal than everyday sounds (as used in auditory icons) or musical motifs (as used in earcons). He put the counter-argument thus: "everyday sounds vary for different cultures anyway, as do the ways of constructing musical motifs", so there is no reason in principle why these types of acoustic ecology cannot be successfully used in the right context. Cohen also identified the importance of allowing users to assign their own choice of sound sets to auditory monitoring applications: this catering for user preference is another principle of aesthetic computing. Indeed, Cohen suggested that users could "choose familiar genres based on aesthetic preference" [31].

For their WebMelody system (see section 18.5.1) Barra et al. [5] drew upon the ideas of futurist composer Luigi Russolo (1885-1947), principles of Pierre Schaeffer's *musique concrète*, and found inspiration in Edgard Varèse's *Poème Electronique* (1958) and John Cage's aleatoric compositions, to construct sonifications that were "neutral with respect to the usual and conventional musical themes".[18] In other words, they attempted to move away from the idioms of tonal and atonal music and towards the more abstract syntaxes found in the electroacoustic/*musique concrète*/organized sound traditions. *Musique concrète* approaches composition not by writing a tune which is then given to players to render in sound but instead by first recording existing or 'found' (concrete) sounds and assembling them into a musical piece.[19] It is not hard to see how auditory-icon-based approaches might fit into this mould given the auditory icon's origin in existing real-world sound. Vickers sounded a note of caution at such moves away from tonal music systems [116]:

> In the pursuit of aesthetic excellence we must be careful not to tip the balance too far in favour of artistic form. Much current art music would not be appropriate for a sonification system. The vernacular is popular music, the aesthetics of which

[17] The role of sonification in aesthetic computing has not gone unnoticed by Fishwick though as evidenced by the inclusion of a chapter by Vickers and Alty [124] on program sonification in the volume *Aesthetic Computing* [45]. The relationship between sonification design and aesthetics is discussed more fully in Chapter 7 of this volume.

[18] For example, *Music of Changes* (1951) and much of his music for prepared piano.

[19] This is in contrast to *musique abstraite* which is what traditional compositional techniques produce.

> are often far removed from the ideals of the music theorists and experimentalists. [p. 6]

This admonition was posited on the observations by Lucas [79] that the recognition accuracy of an auditory display was increased when users were made aware of the display's musical design principles. Furthermore, Watkins and Dyson [126] demonstrated that melodies following the rules of western tonal music are easier to learn, organize cognitively, and discriminate than control tone sequences of similar complexity. This, Vickers argued, suggested that the cognitive organizational overhead associated with atonal systems makes them less well suited as carriers of program information. However, this reasoning fails to account for the fact that electroacoustic music, whilst often lacking discernible melodies and harmonic structures, is still much easier to organize and decompose cognitively than atonal pieces [117]. The studies of Lucas [79] and Watkins and Dyson [126] were rooted in the tonal/atonal dichotomy and did not consider this other branch of music practice. Vickers's revised opinion [117] that the electroacoustic/*musique concrète*/organized sound traditions can, in fact, offer much to sonification design supports the position taken by Barra et al. [5] in their WebMelody web server sonification system. Indeed, Barra et al. avoided the use of harmonic tonal sequences and rhythmic references because they believed that such structures might distract users by drawing upon their individual "mnemonic and musical (personal) capabilities" [5]. Rather, they "let the sonification's timbre and duration represent the information and avoid recognizable musical patterns" [5, p. 35]. This, they said

> . . . makes it possible to hear the music for a long time without inducing mental and musical fatigue that could result from repeated musical patterns that require a finite listening time and not, as in our case, a potentially infinite number of repetitions.

The system was evaluated in a dual-task experiment in which participants were given a primary text editing task and a secondary web server monitoring task. Here, the monitoring was peripheral because the information provided by the sonification was peripheral. The results indicated that the background music generated by WebMelody did not distract users from their primary task while at the same time alllowing meaningful information to be drawn from it. What is particularly encouraging about this study is that the sonification of web server data was far less distracting than similar visual displays. In a visual-only study, Maglio and Campbell [81] asked participants to perform a text-editing task whilst visually monitoring another process. Not suprisingly (given the need to keep switching visual focus) Maglio and Campbell observed a decrease in participants' performance on the text editing task. This suggests that, when designed well, a process monitoring sonification can provide a useful, minimally-distracting data display in situations where peripheral-monitoring is required.

When developing their Audio Aura system (a serendipitous-peripheral monitoring system to allow people to have background awareness of an office environment), Mynatt et al. [86, 85] created four separate sets of auditory cues which they called *ecologies*. The reason for the ecology label is that all the sounds within each set were designed to be compatible with the others not just in terms of frequency and intensity balance but in logical terms too. For example, their 'sound effects world' ecology was based around the noises to be heard at the beach: gull cries were mapped to quantities of incoming email with surf and wave noises representing the activity level of members of a particular group. Thus, each sound in a

particular ecology would not sound out of place with the others. In all, four ecologies were constructed:

1. **Voice world** — vocal speech labels;
2. **Sound effects world** — beach noises: an auditory icon/soundscape set;
3. **Music world** — tonal musical motifs: a structured earcon set;
4. **Rich world** — a composite set of musical motifs, sound effects, and vocal messages.

Unfortunately, no formal studies have been published to discover how well the ecologies worked and which of the four was better received by users. In theory, this selection of different ecologies allows user preference to be catered for which is an important principle in aesthetic computing [43, 45]. Such principles can also be found in Tran and Mynatt's Music Monitor [113] which allowed the user to personalize the system by specifying their preferred music tracks upon which the main earcon messages were overlaid.

18.6.4 Comprehensibility and audibility

The audibility of sonifications is an important factor and is tightly coupled to the issue of intrusiveness (see section 18.6.1 above). The comprehensibility of sonifications depends on many factors including the production quality of the sounds, the quality of the playback system, and cultural and metaphoric associations. Many process data require metaphoric or analogic mappings for audio representation as they do not naturally possess their own sound. The choice of metaphor may determine how learnable and comprehensible the mapping is. For example, Kilander and Lönnqvist found that their sound of a golf ball dropping into a cup was difficult for listeners to recognize "except possibly for avid golfers" whilst the sound of a car engine was easy to identify [75]. This highlights the fact that when using real-world sounds it is important to assess the cultural attributes of those sounds. Investigating musical tones for the monitoring of background processes Søråsen [106] found that sudden onset or disappearance of a timbre is easier to detect than changes in the rhythm and melody of that timbre. He concluded "changes within one single instrument should be very carefully designed to represent non-binary changes in state or modus".

18.7 The road ahead

The above discussion has outlined several pitfalls that the researcher wishing to develop auditory display solutions for monitoring activities may face. The remaining sections discuss several strategies and techniques that look promising for avoiding these problems.

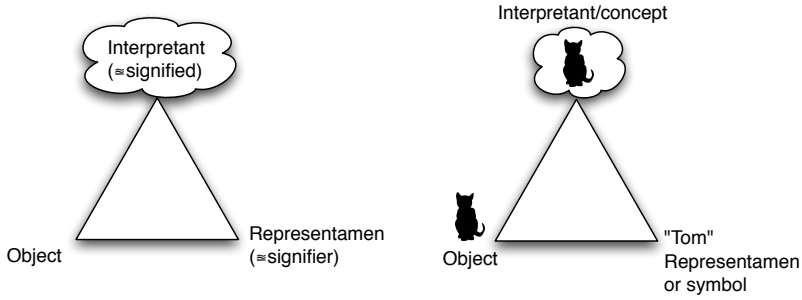
18.7.1 Representation and meaning making

Kramer portrayed auditory display as a representational continuum ranging from analogic to symbolic mappings [77, pp. 22–29]. Analogic representations are those which have an instrinsic correspondence to the data and a good example would be the field of audification (see Chapter 12). Symbolic representations are indirect, possibly involving abstractions or amalgamations of one or more data sets and much (if not most) sonification work lies towards

this end of Kramer's continuum. Rauterberg and Styger [91] recommended the use of iconic sound mappings for real-time direct and peripheral process monitoring. They said that we should "look for everyday sounds that 'stand for themselves'". The question of representation is important and is richer than a simple anaolgic/symbolic continuum. Semiotics offers us the concept of *sign*. Signs are words, images, sounds, smells, objects, etc. that have no intrinsic meaning and which become signs when we attribute meaning to them [29]. Signs stand for or represent something beyond themselves. Modern semiotics is based upon the work of two principal thinkers, the linguist Ferdinand de Saussure and the philosopher and logician Charles Sanders Peirce. In Saussurean semiotics the sign is a dyadic relationship between the semiotics!signifiersignifier and the *signified*. The signifier represents the signified and both are psychological constructs, so Saussure's signifier has no real-world referent (though modern day interpretations have the signifier taking a more material form). The spoken word 'tree', for example, is the signifier for the concept of the thing we know as a tree. The sign thus formed is a link between the sound pattern and the concept. Peirce's semiotics is based upon a triadic relationship comprising:

- The *object*: a real-world referent, the thing to be represented (note, this need not have a material form);
- The *representamen*: the form the sign takes (word, image, sound, etc.,) and which represents the object);
- The *interpretant*: the sense we make of the sign.

Figure 18.8 shows two Peircean triads drawn as 'meaning triangles' (after Sowa [107]). It should be noted that the Saussurean signifier and signified correspond only approximately to Peirce's representamen and interpretant. Figure 8(a) shows the basic structure of a Peircean sign and Figure 8(b) shows the sign formed by the name Tom which represents a specific individual cat with that name.



(a) A Peircean semiotic triad (after Sowa [107]). Approximations to de Saussure's semiotic terminology given in parentheses.

(b) Example: A real individual cat is the object. It is signified by the symbol "Tom" which brings into our mind the concept of Tom the cat.

Figure 18.8: Two semiotic 'meaning triangles'

To relate this to visualization consider Figure 18.9 which shows a semiotic relationship that exists in a spreadsheet application. The spreadsheet program takes a data set (in this case student grades) which has been collected from the real world of a cohort of students studying

a course. These data are then presented to the user via the tabular visual representation we would be familiar with when launching our favourite spreadsheet program. It should be noted that this visual presentation is not the data set itself, but a particular representation of it. The tabular view, then, becomes the representamen of the data. The interpretant in this sign is the sense we make of the student marks by looking at the screen.

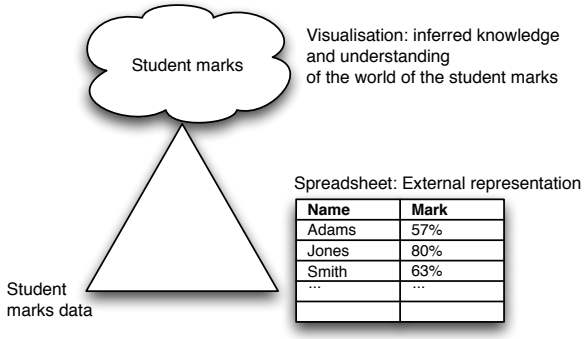


Figure 18.9: The common spreadsheet display is an external representation of the underlying data. The interpretant is the concept that is formed in our mind when we view the tabular representation.

Semiotics offers us three modes of representation: symbolic, iconic, and indexical. Symbolic signs are purely arbitrary (e.g., a "no entry" sign on a road). Iconic signs resemble the object in some way and this mode could include metaphors. Indexical signs are directly connected to the object (e.g., a rash as a sign of a particular disease). Thus Kramer's continuum distinguishes between indexical representations at its analogic end and symbolic representations at the other; iconic representations are not explicitly covered. It will be helpful in future work to consider representations from a semiotic perspective as it helps to clarify that central region of Kramer's continuum. For instance, earcons can be metaphoric but some are more symbolic than others. Consider Brewster's early example [18] which had a purely arbitary earcon hierarchy to represent various operating system events. Contrast this with the CAITLIN system [121] in which the sounds were metaphoric: programming language loops were represented by a repeating motif, selections by an up-and-down motif which resembles the intonation patterns of the human voice when asking and answering a question. The best example of an indexical auditory display would be audifications (e.g., Hayward's seismograms [58]) as the sounds are directly related to the data (the semiotic 'object'). In sonification we are primarily working with symbolic and iconic mappings. Rauterberg and Styger [91] suggested that iconic mappings are particularly well-suited to monitoring applications, but work in network and web server monitoring also indicates that symbolic mappings (e.g., packet size mapped to the sound of water) can also work well when combined in such a way as to provide a non-distracting soundscape. What is needed are studies that are designed specifically to explore the impact of iconic, symbolic, and indexical mappings in the three monitoring modes (direct, peripheral, and serendipitous-peripheral). Systematic investigation using these explicit classifications will provide a much clearer body of knowledge and evidence than is currently available.

18.7.2 Aesthetics

In their work on the ARKola system Gaver et al. [54] suggested that auditory icons had less potential for annoyance than musical messages because they can be designed to complement the auditory environment in which they will be used. This argument is influenced by principles found in acoustic ecology. Because they use real-world sounds it is said that auditory icons do not just sit in an acoustic environment but can also extend it in a complementary manner [54] unlike musical messages which are naturally alien to the acoustic ecology of the environment. However, a note of caution should be sounded at this point. Acoustic ecology as a field of study was first defined by R. Murray Schafer in his 1977 book "The Tuning of the World" [100] which came out of his earlier work on soundscapes. Schafer encouraged people to hear the acoustic environment as a musical composition for which the listener should own the responsibility of its composition (see Wrightson [128]). That Schafer sees ecologies of real-world sounds as musical compositions shows that the distinction between musical and non-musical sound is not as clear as proposed by Gaver et al. Indeed, Varèse referred to some of his electroacoustic music as *organized sound* (e.g., *Poème Electronique* (1965)); in the electroacoustic/organized sound/*musique concrète* schemes there is no requirement for the music to possess melodic components in the normal sense of tonal and atonal music schemes. Indeed, it could be argued that all sonification can be viewed (type cast) as music if the sonification is seen from the electroacoustic weltanschauung [125, 117]. What Gaver et al. are suggesting is that melodic motif-based sonifications are potentially more annoying than those using natural or real-world sounds. It is not clear from the evidence available that this is so. Indeed, Barra et al. [5] and Tran and Mynatt [113] used musical components to great effect in their peripheral and serendipitous-peripheral monitoring work. Tractinsky et al. [112] found that aesthetics plays a very strong role in user perception and there is a further growing body of evidence that systems which are designed with a conscious attention to the aesthetic dimension benefit from increase usability. In monitoring situations in which the auditory output needs to be continuous or frequent this dimension is especially important to ensure the avoidance of user annoyance and fatigue. The relationship between sonification and aesthetics and the art vs. science debate are discussed in more detail in Chapter 7.

Semiotics and aesthetics are two broad areas that provide the language for talking about representational schemata and which give us general theoretical foundations to inform the development of future sonification systems. In terms of the narrower problem of process monitoring and its attendant pitfalls several specific sound design techniques have emerged that offer potential as successful sonification strategies and which are worthy of further more focused research: soundscapes, steganography, model-based sonification, and spatialization. These are discussed briefly below.

18.7.3 Soundscapes

In the research discussed in the preceding sections some of the most successful results came from projects that delivered the sonifications through soundscapes or sound frameworks based upon soundscape principles. Many of these soundscapes are characterized in part by the use of natural real-world sounds (rather than musical instruments) containing large amounts of (modulated and filtered) broadband noise.

Cohen [32] claimed his acoustic ecology was unobtrusive meaning that problems with

annoyance and distraction would be less than with other sonification designs and more recent research (e.g., Kainulainen et al. [73] and Jung [72]) seems to back up these early findings. At the same time new techniques have begun to emerge for studying how people interact with and understand soundscapes, and the stages of learning they pass through. For example, Droumeva and Wakkary [39] devised the concept of "aural fluency" which has three stages marking progression from familiarisation with the logic and syntax of a soundscape through to becoming fluent in the soundscape's language. With such tools it is now much easier to design detailed studies to investigate the issues involved in the production of soundscape-based auditory displays for monitoring tasks.

18.7.4 Steganographic embedding

Another technique that has led to increased user satisfaction and lower annoyance and distraction levels is the embedding of signal sounds inside some carrier sound. In the computer security and encryption worlds this would be called *steganography* (literally "concealed writing"). Tran and Mynatt's Music Monitor system [113] used just such an approach whereby the sonification signals were overlaid on a piece of user-selected music. Jung [72] describes a similar strategy for a user notification system. For tasks requiring user notification Butz and Jung [25] confirmed that auditory cues (in this case musical motifs) could be effectively embedded in an ambient soundscape. Such a steganographic approach allows monitoring to be carried out by those who need to do it without causing distraction to other people in the environment. As the Tran and Mynatt and Butz and Jung examples show, message embedding can be done with music or soundscapes alike and so this is another direction future research into sonification for process monitoring should explore. (Note, this sort of embedding is different from the piggy-backing of variables onto a single complex tone that Fitch and Kramer [46] describe.)

18.7.5 Physical modeling and model-based sonification

Going beyond direct parameter mapping, researchers have used physical models to allow more complex bindings between data and sound. Typically, a sound generating model (from a simple resonator to a multi-parameter model) is created in an appropriate audio environment or programming language (e.g., SuperCollider, Pure Data, Max/MSP, etc.) and then the data to be monitored is used to excite or perturb the model. For example, Ballora et al. used this technique successfully for sonifying network traffic [4]. We have seen that the majority of the soundscape approaches to process monitoring relied on pre-recorded audio files such that incoming data would trigger the playback of a discrete sound. Physical modeling offers the potential for increased expressivity because it allows more than just the amplitude or pitch of a sound to be controlled; slight changes in data would lead to changes in the continuous soundscape.

Model-based sonification takes this idea and turns it on its head so that the data set becomes the sound model (the resonator) and the user is left to manually excite or perturb the model to infer knowledge about the data. However, there is no reason why the excitation has to be manual (e.g., see Hermann and Ritter [61]). Indeed, Chafe et al. [27, 28] explored network sonification by setting the network state as the sound model and letting incoming

ping messages 'play' the model. They offer some useful insight into how to create such sound models and so this is another avenue of research that should be further explored.

18.7.6 Spatialization and HRTFs

Spatialization of audio has also been shown to be important in user performance in real-time monitoring tasks. Roginska et al. [92] suggest the use of head-related transfer functions (HRTF) to ensure highest performance and Best et al. [9] found that spatial segregation of sound sources using HRTFs is advantageous when engaged in divided attention tasks.[20] In both pieces of research it was the spatialization that HRTFs afford that led to the performance increase. Of course, HRTFs are computationally expensive to produce and require headphones in use and so are not suitable for all monitoring scenarios (especially in shared environments where more than one person may be involved in the monitoring). However, where they are appropriate to use they work very well and will benefit from further research.

18.7.7 Closing remarks

Since work on auditory display and sonification began in earnest, the monitoring of environments and background processes has consistently attracted attention. Perhaps this is because audio seemingly provides a natural medium for this type of information. Process and environmental data are often supplementary to other primary task data and the affordances that sound offers through being able to occupy space in the perceptual background could be used to great effect. Despite the many different applications and approaches, common themes of dealing with intrusiveness, annoyance, and comprehensibility have risen to the fore. Furthermore, these themes are often linked to a common thread, the aesthetic design of the sonifications. Researchers who reported the most success also dealt directly with the issue of the aesthetics and acoustic ecology of their sonifications. It is suggested that the agenda for research in this field of sonification should be underpinned by a conscious attention to the role of semiotics and aesthetics and that these foundations should be used in conjunction with techniques involving soundscapes, steganographic embedding, model-based sonification, and spatialization to develop the next generation of real-time and process monitoring sonification applications. Chapter 7 in this volume presents a more detailed treatment of the relationship between sonification design and aesthetics.

Bibliography

[1] James L. Alty. Can we use music in computer-human communication? In M. A. R. Kirby, A. J. Dix, and J. E. Finlay, editors, *People and Computers X: Proceedings of HCI '95*, pages 409–423. Cambridge University Press, Cambridge, 1995.

[20] The HRTF uses mathematical transformations to produce realistic spatial audio by converting a recorded sound signal "to that which would have been heard by the listener if it had been played at the source location, with the listener's ear at the receiver location" (see http://en.wikipedia.org/wiki/Head-related_transfer_function). This results in highly accurate spatial reproduction of sound offering not just left-right discrimination of a sound's source in space but also front-back and up-down. Because every person's auditory system is unique (thanks, in part, to different shaped ears) to get an accurate spatial representation of a sound, the HRTF needs to be computed individually. Averaged HRTFs have been produced for general use, but their perceived accuracy depends on the closeness of match between the listener's auditory system and the HRTF.

[2] Ronald M. Baecker and John Buchanan. A programmer's interface: A visually enhanced and animated programming environment. In *Twenty-Third Annual Hawaii International Conference on Systems Sciences*, volume 11, pages 531–540. IEEE Computer Society Press, 1990.

[3] Saskia Bakker, Elise van den Hoven, and Berry Eggen. Exploring interactive systems using peripheral sounds. In Rolf Nordahl, Stefania Serafin, Federico Fontana, and Stephen Brewster, editors, *Haptic and Audio Interaction Design*, volume 6306 of *Lecture Notes in Computer Science*, pages 55–64. Springer Berlin / Heidelberg, 2010.

[4] Mark Ballora, Brian Panulla, Matthew Gourley, and David L. Hall. Preliminary steps in sonifying web log data. In Eoin Brazil, editor, *16th International Conference on Auditory Display*, pages 83–87, Washington, DC, 9–15 June 2010. ICAD.

[5] Maria Barra, Tania Cillo, Antonio De Santis, Umberto F. Petrillo, Alberto Negro, and Vittorio Scarano. Multimodal monitoring of web servers. *IEEE Multimedia*, 9(3):32–41, 2002.

[6] Maria Barra, Tania Cillo, Antonio De Santis, Umberto Ferraro Petrillo, Alberto Negro, and Vittorio Scarano. Personal WebMelody: Customized sonification of web servers. In Jarmo Hiipakka, Nick Zacharov, and Tapio Takala, editors, *Proceedings of the 2001 International Conference on Auditory Display*, pages 1–9, Espoo, Finland, 29 July–1 August 2001. ICAD.

[7] Lewis I. Berman and Keith B. Gallagher. Listening to program slices. In Tony Stockman, Louise Valgerður Nickerson, Christopher Frauenberger, Alistair D. N. Edwards, and Derek Brock, editors, *ICAD 2006 - The 12th Meeting of the International Conference on Auditory Display*, pages 172–175, London, UK, 20–23 June 2006. Department of Computer Science, Queen Mary, University of London, UK.

[8] Lewis I. Berman and Keith B. Gallagher. Using sound to understand software architecture. In *SIGDOC '09: Proceedings of the 27th ACM international conference on Design of communication*, pages 127–134, New York, NY, USA, 2009. ACM.

[9] Virginia Best, Antje Ihlefeld, and Barbara Shinn-Cunningham. The effect of auditory spatial layout in a divided attention task. In Eoin Brazil, editor, *ICAD 2005 - The 11th Meeting of the International Conference on Auditory Display*, pages 17–22, Limerick, Ireland, 6–9 July 2005.

[10] David B. Boardman, Geoffrey Greene, Vivek Khandelwal, and Aditya P. Mathur. LISTEN: A tool to investigate the use of sound for the analysis of program behaviour. In *19th International Computer Software and Applications Conference*, Dallas, TX, 1995. IEEE.

[11] David B. Boardman and Aditya P. Mathur. Preliminary report on design rationale, syntax, and semantics of LSL: A specification language for program auralization. Technical report, Dept. of Computer Sciences, Purdue University, 21 September 1993.

[12] Dale S. Bock. ADSL: An auditory domain specification language for program auralization. In Gregory Kramer and Stuart Smith, editors, *ICAD '94 Second International Conference on Auditory Display*, pages 251–256, Santa Fe, NM, 1994. Santa Fe Institute.

[13] Dale S. Bock. *Auditory Software Fault Diagnosis Using a Sound Domain Specification Language*. Ph.D. thesis, Syracuse University, 1995.

[14] Dale S. Bock. Sound enhanced visualization: A design approach based on natural paradigms. M.sc. dissertation, Syracuse University, 1995.

[15] Till Bovermann. *Tangible Auditory Interfaces: Combining Auditory Displays and Tangible Interfaces*. Ph.D. thesis, Bielefeld University, Germany, 15 December 2010.

[16] Till Bovermann, René Tünnermann, and Thomas Hermann. Auditory augmentation. *International Journal of Ambient Computing and Intelligence*, 2(2):27–41, 2010.

[17] Stephen Brewster. Navigating telephone-based interfaces with earcons. In H Thimbleby, B O'Conaill, and P Thomas, editors, *People and Computers XII*, pages 39–56. Springer, Berlin, 1997.

[18] Stephen Brewster. The design of sonically-enhanced widgets. *Interacting with Computers*, 11:211–235, 1998.

[19] Stephen A. Brewster. A sonically-enhanced interface toolkit. In Steven P. Frysinger and Gregory Kramer, editors, *ICAD '96 Third International Conference on Auditory Display*, pages 47–50, Palo Alto, 1996. Xerox PARC, Palo Alto, CA 94304.

[20] Stephen A. Brewster. Using earcons to improve the usability of tool palettes. In *CHI '98: CHI 98 conference summary on Human factors in computing systems*, pages 297–298, New York, NY, USA, 1998. ACM Press.

[21] Stephen A. Brewster. Overcoming the lack of screen space on mobile computers. Department of computing science technical report, Glasgow University, April 2001.

[22] Stephen A. Brewster, Veli-Pekka Raty, and Atte Kortekangas. Enhancing scanning input with non-speech sounds. In *ASSETS 96*, pages 10–14, Vancouver, BC, Canada, 1996. ACM.

[23] Stephen A. Brewster, Peter C. Wright, and Alistair D. N. Edwards. A detailed investigation into the effectiveness of earcons. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 471–498. Addison-Wesley, Reading, MA, 1994.

[24] Marc H. Brown and John Hershberger. Color and sound in algorithm animation. *Computer*, 25(12):52–63, 1992.

[25] Andreas Butz and Ralf Jung. Seamless user notification in ambient soundscapes. In *Proceedings of the 10th international conference on Intelligent user interfaces*, IUI '05, pages 320–322, New York, NY, USA, 2005. ACM.

[26] William Buxton. Introduction to this special issue on nonspeech audio. *Human-Computer Interaction*, 4:1–9, 1989.

[27] Chris Chafe and Randal Leistikow. Levels of temporal resolution in sonification of network performance. In Jarmo Hiipakka, Nick Zacharov, and Tapio Takala, editors, *ICAD 2001 7th International Conference on Auditory Display*, pages 50–55, Espoo, Finland, 29 July–1 August 2001. ICAD.

[28] Chris Chafe, Scott Wilson, and Daniel Walling. Physical model synthesis with application to internet acoustics. In *Proceedings of ICASSP02: IEEE International Conference on Acoustics, Speech and Signal Processing*, Orlando, Florida, 13–17 May 2002.

[29] Daniel Chandler. *Semiotics: The Basics*. Routledge, 2 edition, 2007.

[30] Michel Chion. *Audio-Vision: Sound on Screen*. Columbia University Press, NY, 1994.

[31] Jonathan Cohen. "Kirk Here": Using genre sounds to monitor background activity. In S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel, and T. White, editors, *INTERCHI '93 Adjunct Proceedings*, pages 63–64. ACM, New York, 1993.

[32] Jonathan Cohen. Monitoring background activities. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 499–532. Addison-Wesley, Reading, MA, 1994.

[33] Jonathan Cohen. OutToLunch: Further adventures monitoring background activity. In Gregory Kramer and Stuart Smith, editors, *ICAD '94 Second International Conference on Auditory Display*, pages 15–20, Santa Fe, NM, 1994. Santa Fe Institute.

[34] Alex Paul Conn. Time affordances: The time factor in diagnostic usability heuristics. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 186–193, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[35] Murray Crease and Stephen A. Brewster. Scope for progress — monitoring background tasks with sound. In *Proc. INTERACT '99*, volume II, pages 19–20, Edinburgh, UK, 1999. British Computer Society.

[36] Murray G. Crease and Stephen A. Brewster. Making progress with sounds — the design and evaluation of an audio progress bar. In S. A. Brewster and A. D. N. Edwards, editors, *ICAD '98 Fifth International Conference on Auditory Display*, Electronic Workshops in Computing, Glasgow, 1998. British Computer Society.

[37] Christopher J. DiGiano. *Visualizing Program Behaviour Using Non-Speech Audio*. M.sc. dissertation, University of Toronto, 1992.

[38] Christopher J. DiGiano and Ronald M. Baecker. Program auralization: Sound enhancements to the programming environment. In *Graphics Interface '92*, pages 44–52, 1992.

[39] Milena Droumeva and Ron Wakkary. socio-ec(h)o: Focus, listening and collaboration in the experience of ambient intelligent environments. In Eoin Brazil, editor, *16th International Conference on Auditory Display*, pages 327–334, Washington, DC, 9–15 June 2010. ICAD.

[40] Berry Eggen, Koert van Mensvoort, David Menting, Emar Vegt, Wouter Widdershoven, and Rob Zimmermann. Soundscapes at Workspace Zero —- design explorations into the use of sound in a shared environment. In *Pervasive '08 Sixth International Conference on Pervasive Computing*, 19–22 May 2008.

[41] J. Louise Finlayson and Chris Mellish. The 'AudioView' — providing a glance at Java source code. In Eoin Brazil, editor, *ICAD 2005 - The 11th Meeting of the International Conference on Auditory Display*, pages 127–133, Limerick, Ireland, 6–9 July 2005.

[42] J. Louise Finlayson, Chris Mellish, and Judith Masthoff. Fisheye views of Java source code: An updated LOD algorithm. In Constantine Stephanidis, editor, *Universal Access in Human-Computer Interaction. Applications and Services*, volume 4556 of *Lecture Notes in Computer Science*, pages 289–298. Springer Berlin / Heidelberg, 2007.

[43] Paul Fishwick. Aesthetic programming: Crafting personalized software. *Leonardo*, 35(4):383–390, 2002.

[44] Paul A. Fishwick. Aesthetic computing manifesto. *Leonardo*, 36(4):255, September 2003.

[45] Paul A. Fishwick, editor. *Aesthetic Computing*. MIT Press, April 2006.

[46] W. Tecumseh Fitch and Gregory Kramer. Sonifying the body electric: Superiority of an auditory over a visual display in a complex, multivariate system. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 307–326. Addison-Wesley, Reading, MA, 1994.

[47] Joan M. Francioni, Larry Albright, and Jay Alan Jackson. Debugging parallel programs using sound. *SIGPLAN Notices*, 26(12):68–75, 1991.

[48] Joan M. Francioni, Jay Alan Jackson, and Larry Albright. The sounds of parallel programs. In *6th Distributed Memory Computing Conference*, pages 570–577, Portland, Oregon, 1991.

[49] Joan M. Francioni and Diane T. Rover. Visual-aural representations of performance for a scalable application program. In *High Performance Computing Conference*, pages 433–440, 1992.

[50] William Gaver, Thomas Moran, Allan MacLean, Lennart Lövstrand, Paul Dourish, Kathleen Carter, and William Buxton. Realizing a video environment: Europarc's rave system. In P. Bauersfeld, J. Bennett, and G. Lynch, editors, *CHI '92 Conference on Human Factors in Computing Systems*, pages 27–35, Monterey, CA, 1992. ACM Press/Addison-Wesley.

[51] William W. Gaver. Auditory icons: Using sound in computer interfaces. *Human Computer Interaction*, 2:167–177, 1986.

[52] William W. Gaver. *Everyday Listening and Auditory Icons*. Ph.D. thesis, University of California, San Diego, 1988.

[53] William W. Gaver. The SonicFinder: An interface that uses auditory icons. *Human Computer Interaction*, 4(1):67–94, 1989.

[54] William W. Gaver, Randall B. Smith, and Tim O'Shea. Effective sounds in complex systems: The arkola simulation. In S. Robertson, G. Olson, and J Olson, editors, *CHI '91 Conference on Human Factors in Computing Systems*, pages 85–90, New Orleans, 1991. ACM Press/Addison-Wesley.

[55] Michael Gilfix and Alva L. Couch. Peep (the network auralizer): Monitoring your network with sound. In *14th System Administration Conference (LISA 2000)*, pages 109–117, New Orleans, Louisiana, USA, 3–8 December 2000. The USENIX Association.

[56] M. C. Gopinath. Auralization of intrusion detection system using JListen. Master's thesis, Birla Institute of Technology and Science, Pilani (Rajasthan), India, May 2004.

[57] Carl Gutwin and Saul Greenberg. Support for group awareness in real time desktop conferences. In *Proceedings of The Second New Zealand Computer Science Research Students' Conference*, Hamilton, New Zealand, 18–21 April 1995. University of Waikato.

[58] Chris Hayward. Listening to the Earth sing. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 369–404. Addison-Wesley, Reading, MA, 1994.

[59] Thomas Hermann, Jan M. Drees, and Helge Ritter. Broadcasting auditory weather reports – a pilot project. In Eoin Brazil and Barbara Shinn-Cunningham, editors, *ICAD '03 9th International Conference on Auditory*

Display, pages 208–211, Boston, MA, 2003. ICAD.

[60] Thomas Hermann and Helge Ritter. Listen to your data: Model-based sonification for data analysis. In G. E. Lasker, editor, *Advances in intelligent computing and multimedia systems*, pages 189–194, Baden-Baden, Germany, August 1999. Int. Inst. for Advanced Studies in System research and cybernetics.

[61] Thomas Hermann and Helge Ritter. Crystallization sonification of high-dimensional datasets. *ACM Transactions on Applied Perception*, 2(4):550–558, October 2005.

[62] Scott E. Hudson and Ian Smith. Techniques for addressing fundamental privacy and disruption tradeoffs in awareness support systems. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 248–257, New York, NY, USA, 1996. ACM Press.

[63] Khaled Hussein, Eli Tilevich, Ivica Ico Bukvic, and SooBeen Kim. Sonification design guidelines to enhance program comprehension. In *ICPC '09 IEEE 17th International Conference on Program Comprehension*, pages 120–129, 2009.

[64] Jay Alan Jackson and Joan M. Francioni. Aural signatures of parallel programs. In *Twenty-Fifth Hawaii International Conference on System Sciences*, pages 218–229, 1992.

[65] Jay Alan Jackson and Joan M. Francioni. Synchronization of visual and aural parallel program performance data. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 291–306. Addison-Wesley, Reading, MA, 1994.

[66] Michael A. Jackson. *System Development*. Prentice-Hall International, 1983.

[67] David H. Jameson. The run-time components of Sonnet. In Gregory Kramer and Stuart Smith, editors, *ICAD '94 Second International Conference on Auditory Display*, pages 241–250, Santa Fe, NM, 1994. Santa Fe Institute.

[68] David H. Jameson. Sonnet: Audio-enhanced monitoring and debugging. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 253–265. Addison-Wesley, Reading, MA, 1994.

[69] James J. Jenkins. Acoustic information for object, places, and events. In W. H. Warren, editor, *Persistence and Change: Proceedings of the First International Conference on Event Perception*, pages 115–138. Lawrence Erlbaum, Hillsdale, NJ, 1985.

[70] Martin Jonsson, Calle Jansson, Peter Lönnqvist, Patrik Werle, and Fredrik Kilander. Achieving non-intrusive environments for local collaboration. Technical Report FEEL Project Deliverable 2.2(1), Department of Computer and Systems Sciences, IT University, Kista, 2002.

[71] Abigail J. Joseph and Suresh K. Lodha. Musart: Musical audio transfer function real-time toolkit. In *ICAD '02 - 2002 International Conference on Auditory Display*, Kyoto, Japan, 2002.

[72] Ralf Jung. Ambience for auditory displays: Embedded musical instruments as peripheral audio cues. In Brian Katz, editor, *Proc. 14th Int. Conf. Auditory Display (ICAD 2008)*, Paris, France, 24–27 June 2008. ICAD.

[73] Anssi Kainulainen, Markku Turunen, and Jaakko Hakulinen. An architecture for presenting auditory awareness information in pervasive computing environments. In Tony Stockman, Louise Valgerður Nickerson, Christopher Frauenberger, Alistair D. N. Edwards, and Derek Brock, editors, *ICAD 2006 - The 12th Meeting of the International Conference on Auditory Display*, pages 121–128, London, UK, 20–23 June 2006.

[74] Frank Kilander and Peter Lönnqvist. A whisper in the woods – an ambient soundscape for peripheral awareness of remote processes. In *Continuity in Future Computing Systems Workshop I3*, Spring Days, Porto, Portugal, 2001.

[75] Frank Kilander and Peter Lönnqvist. A whisper in the woods – an ambient soundscape for peripheral awareness of remote processes. In *ICAD 2002 – International Conference on Auditory Display*, 2002.

[76] Donald E. Knuth. BCS/IET Turing lecture, February 2011.

[77] Gregory Kramer. An introduction to auditory display. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 1–78. Addison-Wesley, Reading, MA, 1994.

[78] Tami Lapidot and Orit Hazzan. Song debugging: Merging content and pedagogy in computer science

education. *SIGCSE Bull.*, 37:79–83, December 2005.

[79] Paul A. Lucas. An evaluation of the communicative ability of auditory icons and earcons. In *ICAD '94 Second International Conference on Auditory Display*, pages 121–128, Santa Fe, NM, 1994. Santa Fe Institute.

[80] Janet Lumsden, Stephen A. Brewster, Murray Crease, and Philip D. Gray. Guidelines for audio-enhancement of graphical user interface widgets. In *Proceedings of British HCI 2002*, volume II, pages 6–9. BCS, London, UK, September 2002.

[81] Paul P. Maglio and Christopher S. Campbell. Tradeoffs in displaying peripheral information. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 241–248, New York, NY, USA, 2000. ACM Press.

[82] Aditya P. Mathur, David B. Boardman, and Vivek Khandelwal. LSL: A specification language for program auralization. In Gregory Kramer and Stuart Smith, editors, *ICAD '94 Second International Conference on Auditory Display*, pages 257–264, Santa Fe, NM, 1994. Santa Fe Institute.

[83] Gottfried Mayer-Kress, Robin Bargar, and Insook Choi. Musical structures in data from chaotic attractors. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 341–368. Addison-Wesley, Reading, MA, 1994.

[84] S. Joy Mountford and William Gaver. Talking and listening to computers. In B. Laurel and S. Mountford, editors, *The Art of Human-Computer Interface Design*, pages 319–334. Addison-Wesley, Reading, MA, 1990.

[85] Elizabeth D. Mynatt, Maribeth Back, Roy Want, Michael Baer, and Jason B. Ellis. Designing Audio Aura. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 566–573, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

[86] Elizabeth D. Mynatt, Maribeth Back, Roy Want, and Ron Frederick. Audio Aura: Light-weight audio augmented reality. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 211–212, New York, NY, USA, 1997. ACM Press.

[87] Elin Rønby Pedersen. People presence or room activity supporting peripheral awareness over distance. In *CHI '98: CHI 98 conference summary on Human factors in computing systems*, pages 283–284, New York, NY, USA, 1998. ACM Press.

[88] Elin Rønby Pedersen and Tomas Sokoler. Aroma: Abstract representation of presence supporting mutual awareness. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 51–58, New York, NY, USA, 1997. ACM Press.

[89] R. Jagadish Prasath. Auralization of web server using JListen. Master's thesis, Birla Institute of Technology and Science, Pilani (Rajasthan), India, May 2004.

[90] Marty Quinn. Seismic Sonata: A musical replay of the 1994 Northridge, California earthquake, 2000.

[91] Matthias Rauterberg and Erich Styger. Positive effects of sound feedback during the operation of a plant simulator. In B. Blumenthal, J. Gornostaev, and C. Unger, editors, *Human Computer Interaction: 4th International Conference, EWHCI '94, St. Petersburg, Russia, August 2 - 5, 1994. Selected Papers*, volume 876 of *Lecture Notes in Computer Science*, pages 35–44. Springer, Berlin, 1994.

[92] Agnieszka Roginska, Edward Childs, and Micah K. Johnson. Monitoring real-time data: A sonification approach. In Tony Stockman, Louise Valgerður Nickerson, Christopher Frauenberger, Alistair D. N. Edwards, and Derek Brock, editors, *ICAD 2006 - The 12th Meeting of the International Conference on Auditory Display*, pages 176–181, London, UK, 20–23 June 2006.

[93] Pablo Romero, Richard Cox, Benedict du Boulay, and Rudi Lutz. A survey of external representations employed in object-oriented programming environments. *Journal of Visual Languages & Computing*, 14(5):387–419, 2003.

[94] Pablo Romero, Richard Cox, Benedict du Boulay, and Rudi K. Lutz. Visual attention and representation switching during Java program debugging: A study using the Restricted Focus Viewer. *Lecture Notes in Computer Science*, 2317:221–235, 2002.

[95] Pablo Romero, Benedict du Boulay, Richard Cox, Rudi Lutz, and Sallyann Bryant. Debugging strategies and tactics in a multi-representation software environment. *International Journal of Human-Computer Studies*,

65(12):992–1009, 2007.

[96] Pablo Romero, Rudi K. Lutz, Richard Cox, and Benedict du Boulay. Co-ordination of multiple external representations during java program debugging. In *IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC'02)*, Arlington, Virginia, USA, 2002. IEEE Computer Society.

[97] Nitin Sawhney and Chris Schmandt. Nomadic radio: Scaleable and contextual notification for wearable audio messaging. In *Proceedings of the CHI 99 Conference on Human factors in computing systems: the CHI is the limit*, pages 96–103, New York, NY, 1999. ACM Press.

[98] Nitin Sawhney and Chris Schmandt. Nomadic radio: Speech and audio interaction for contextual messaging in nomadic environments. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(3):353–383, 2000.

[99] Pierre Schaeffer. *Traité Des Objets Musicaux*. Seuil, Paris, rev. edition, 1967.

[100] R. Murray Schafer. *The Tuning of the World*. Random House, 1977.

[101] Chris Schmandt and Gerardo Vallejo. "ListenIN" to domestic environments from remote locations. In *ICAD '03 9th International Conference on Auditory Display*, pages 22–223, Boston, MA, USA, 6–9 July 2003.

[102] Randall B. Smith. A prototype futuristic technology for distance education. In *Proceedings of the NATO Advanced Workshop on New Directions in Educational Technology*, Cranfield, UK, 10–13 November 1988.

[103] Stuart Smith, Ronald M. Pickett, and Marian G. Williams. Environments for exploring auditory representations of multidimensional data. In Gregory Kramer, editor, *Auditory Display*, volume XVIII of *Santa Fe Institute, Studies in the Sciences of Complexity Proceedings*, pages 167–184. Addison-Wesley, Reading, MA, 1994.

[104] Eric Somers. A pedagogy of creative thinking based on sonification of visual structures and visualization of aural structures. In Stephen A. Brewster and Alistair D. N. Edwards, editors, *ICAD '98 Fifth International Conference on Auditory Display*, Electronic Workshops in Computing, Glasgow, 1998. British Computer Society.

[105] Diane H. Sonnenwald, B. Gopinath, Gary O. Haberman, William M. Keese, III, and John S. Myers. Infosound: An audio aid to program comprehension. In *Twenty-Third Hawaii International Conference on System Sciences*, volume 11, pages 541–546. IEEE Computer Society Press, 1990.

[106] Sigve Søråsen. Monitoring continuous activities with rhythmic music. In *Proceedings of the Student Interaction Design Research Conference*, Sønderborg, 27–28 January 2005. Mads Clausen Institute, University of Southern Denmark.

[107] John F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks/Cole, 2000.

[108] A. Stefik and E. Gellenbeck. Using spoken text to aid debugging: An empirical study. In *ICPC '09 IEEE 17th International Conference on Program Comprehension, 2009.*, pages 110–119, May 2009.

[109] Andreas Stefik, Kelly Fitz, and Roger Alexander. Layered program auralization: Using music to increase runtime program comprehension and debugging effectiveness. In *14th International Conference on Program Comprehension (ICPC 2006)*, pages 89–93, Athens, Greece, 2006. IEEE Computer Society.

[110] Andreas Stefik and Ed Gellenbeck. Empirical studies on programming language stimuli. *Software Quality Journal*, 19(1):65–99, 2011.

[111] Robert S. Tannen. Breaking the sound barrier: Designing auditory displays for global usability. In *4th Conference on Human Factors and the Web*, 5 September 1998.

[112] Noam Tractinsky, Adi Shoval-Katz, and Dror Ikar. What is beautiful is usable. *Interacting with Computers*, 13(2):127–145, 2000.

[113] Quan T. Tran and Elizabeth D. Mynatt. Music monitor: Ambient musical data for the home. In Andy Sloan and Felix van Rijn, editors, *Proceedings of the IFIP WG 9.3 International Conference on Home Oriented Informatics and Telematics (HOIT 2000)*, volume 173 of *IFIP Conference Proceedings*, pages 85–92. Kluwer, 2000.

[114] Gerardo Vallejo. ListenIN: Ambient auditory awareness at remote places. Master's thesis, MIT Media Lab, September 2003.

[115] Paul Vickers. *CAITLIN: Implementation of a Musical Program Auralisation System to Study the Effects on Debugging Tasks as Performed by Novice Pascal Programmers*. Ph.D. thesis, Loughborough University, Loughborough, Leicestershire, September 1999.

[116] Paul Vickers. External auditory representations of programs: Past, present, and future – an aesthetic perspective. In Stephen Barrass and Paul Vickers, editors, *ICAD 2004 – The Tenth Meeting of the International Conference on Auditory Display*, Sydney, 6–9 July 2004. ICAD.

[117] Paul Vickers. Ars informatica – ars electronica: Improving sonification aesthetics. In Luigina Ciolfi, Michael Cooke, Olav Bertelsen, and Liam Bannon, editors, *Understanding and Designing for Aesthetic Experience Workshop at HCI 2005 The 19th British HCI Group Annual Conference*, Edinburgh, Scotland, 2005.

[118] Paul Vickers. Program Auralization: Author's comments on vickers and alty, icad 2000. *ACM Trans. Appl. Percept.*, 2(4):490–494, 2005.

[119] Paul Vickers and James L. Alty. using music to communicate computing information. *Interacting with Computers*, 14(5):435–456, 2002.

[120] Paul Vickers and James L. Alty. musical program auralisation: A structured approach to motif design. *Interacting with Computers*, 14(5):457–485, 2002.

[121] Paul Vickers and James L. Alty. when bugs sing. *Interacting with Computers*, 14(6):793–819, 2002.

[122] Paul Vickers and James L. Alty. Siren songs and swan songs: Debugging with music. *Communications of the ACM*, 46(7):86–92, 2003.

[123] Paul Vickers and James L. Alty. Musical program auralization: Empirical studies. *ACM Trans. Appl. Percept.*, 2(4):477–489, 2005.

[124] Paul Vickers and James L. Alty. The well-tempered compiler: The aesthetics of program auralization. In Paul A. Fishwick, editor, *Aesthetic Computing*, chapter 17, pages 335–354. MIT Press, Boston, MA, 2006.

[125] Paul Vickers and Bennett Hogg. Sonification abstraite/sonification concrète: An 'æsthetic perspective space' for classifying auditory displays in the ars musica domain. In Tony Stockman, Louise Valgerður Nickerson, Christopher Frauenberger, Alistair D. N. Edwards, and Derek Brock, editors, *ICAD 2006 - The 12th Meeting of the International Conference on Auditory Display*, pages 210–216, London, UK, 20–23 June 2006.

[126] Anthony J. Watkins and Mary C. Dyson. On the perceptual organisation of tone sequences and melodies. In Peter Howell, Ian Cross, and Robert West, editors, *Musical Structure and Cognition*, pages 71–119. Academic Press, New York, 1985.

[127] Mark Weiser and John Seely Brown. The coming age of calm technology. 1996.

[128] Kendall Wrightson. An introduction to acoustic ecology. *Soundscape: The Journal of Acoustic Ecology*, 1(1):10–13, Spring 2000.