Measuring the Use of Sound in Everyday Software

Benjamin K. Davison and Bruce N. Walker

Georgia Institute of Technology
Sonification Laboratory
654 Cherry St.
Atlanta, GA, USA 30332-0170
davison@gatech.edu, bruce.walker@psych.gatech.edu

ABSTRACT

Members of the ICAD community might contend that auditory interfaces and even just well-designed sound in computer interfaces could be used more often than is currently the case. However, it is not entirely clear where, when, and how sound is actually being employed in everyday software. We discuss the development of a long-term research project aimed at identifying and categorizing sound use in software. Our mixed-methods approach explores software artifacts from three perspectives: detailed program behavior, source code word count of audio terms, and audio infrastructure. These complementary approaches could provide a deeper understanding of sound use today and, we hope, lead to predicting, guiding, and improving the future trajectory of its use.

1. INTRODUCTION

What is the benefit of audio in everyday computing? We at ICAD demonstrate answers to this question on an annual basis. We show the performance of earcons versus auditory icons and the utility of sound in a visually attentive situation. We study how sound can improve desktop and mobile computer use, and how sonification can enhance data exploration and analysis. These projects suggest a view that sound should be present in computing when it is useful and appropriate.

However, there does not seem to be widespread use of sound in everyday computing.[1] There are certainly some alert sounds, and a ring tone still indicates a phone call, but most applications appear to have few if any sounds. The task of increasing audio interaction begs for a characterization of sound use now. This can highlight difference between use domains and act as a starting point for measuring changes in the characterization data over time.

2. HOW CAN WE INCREASE SOUND USE?

The usefulness of sound is a tacit and fundamental assumption in this auditory display community. However, programmers don't appear to use audio very often [2], and their audio concepts tend to be musical instead of in auditory display or psychoacoustic terminology ("piano" versus "earcon" versus "complex sound") [3]. There has been some effort to reduce the cost of building sound into software [4,5], including an explicit attempt to reduce the cost while maintaining benefits [5]. Toolkit and API[2] improvements provide developer-oriented solutions to the audio use problem. If program authors have easier ways to integrate audio, they may be more likely to use sound in their applications.

Current academic and industry software engineering practice often involves end-user input, providing a way to tailor appropriate auditory interaction to particular scenarios. Even so, the current approach depends heavily on technology changing (hopefully improving) some situation. We are missing an opportunity to understand the bigger picture and broadly relate auditory technology to user groups.

In exploring the development of the modern bicycle, Wiebe Bijker highlights the two aspects of a common representation of the history of technology [6]. First, the concept of a linear development model obscures the rich interrelationships between invention and very gradual adaptation of previous things into new inventions. These details highlight how inventors synthesize previous and parallel inventions into their own work. Second, there is little discussion of the failures along the way. In particular, temporarily popular inventions can help describe what is useful to particular user groups at the time. It is society that creates successful technology, not technology shaping society [6]. Certainly a new technology can change the way people live in the world. But the technology is first adopted because users perceive it to be useful in the existing world.

Bijker demonstrates how the high-wheeled "Ordinary" bicycle provided a way for young, wealthy Europeans to engage in physically risky and public behavior [6]. The solid rubber wheels and a springless frame of the "boneshaker" was an uncomfortable experience. While larger wheels reduced the shakiness, it made mounting and dismounting the bicycle much more difficult. Cautious and unathletic people could not use the ordinary, but it was common among daring young, wealthy men in parts of Europe[3]. The safety and comfort concerns of other potential users motivated further bicycle development which led to the air tire, improved steering and braking, and smaller wheels. The resulting "Safety" bicycle of the early 20th century closely resembles the structure of what passes as a normal bicycle today. From a post-hoc perspective, the high-wheeled Ordinary appears to be a detour on the development of the bicycle; from a social perspective, it satisfied different situation requirements. The Social Construction of Technology [6] drives the engine of innovation.

In computing research, there has been recent concern over the scope of understanding domains and deviating too far from the practice of computing. Dourish discusses user domain exploration and challenges subsequent "implications for design" [8]. He suggests a decoupling of the domain characterization (research in its own right) from programs that are developed with the characterization in mind. Otherwise, the

[1]Some domains of use have frequent auditory interaction. For example, blind computer users have tools such as JAWS [1] that provide a different interactive experience than sighted users have. Identifying what the specific differences *are* motivate this proposed study. *Why* those difference exist is an example of how the data can motivate new research.

[2]Application Programmer Interface. User interface toolkits provide APIs to make it easy to create robust software interface components in fewer lines of software code. See [5].

[3]Understanding non-users can help define the technology as much as understanding the users. See [7] for more on non-users and technology regulation.

characterization may only be useful for the project to justify itself.

Bell and Dourish explored the trajectory of ubiquitous computing, questioning whether fundamental visions of an ubicomp would will be fulfilled. They advocate studying the ubicomp that can be found in a couple of situations today [9]. What could tell the audio community if we are on the right track? A characterization of sound use today could provide a picture of where practice is and a way for domain and temporal comparisons.

In order to make audio-enhanced technologies that "fit", we suggest starting with the characterization of software sound use today. Once there is a better understanding of the social groups, users, technology, and infrastructure it is appropriate to provide an intervention that can best support the entire socio-technical system.

2.1. Use Domains

It appears likely that the use of sound depends somewhat on the domain. The identification of domains is largely guesswork at this point. The literature, discourse analysis (per Bijker [6]), and user studies will further define the important domains. Instead of determining the perfect domains, this task focuses on the sound use in a handful of domains and compares inter-domain results. Through the lens of sound types we can shape our understanding of sound usage in each environmental situation. The domains are broadly defined as individual recreation at home, team computer games, and a white collar office setting.

An office setting is defined as a space with a people who do most of their work with their office computer in a space that is their office or cubicle. The focus will be on which programs large portions of the population use on a regular basis (e.g., a time card system or a mortgage data entry suite). Team gaming situations will focus on the tools that people use to play team games, including the game itself and supporting tools such as communication software. While gaming is typically an at-home event in the United States, it is conducted in social areas in other areas such as Korea [9]. The social aspects of interaction will not be explored in this part of the project, but the source of the data gathering is an important note on the generalizability of findings. Individual recreation at home is increasingly digital and online. This user group is broadly defined as "on the computer" but not involved in a team game or a work situation. This includes individual games such as solitaire, productivity systems such as email or a finance program, and Internet browsing activities such as looking up the news or the latest YouTube videos.

3. CHARACTERIZING SOUND USE TODAY

Exploring software artifacts helps describe both the programmer and the end user. By looking at a program and its affordances, we can discover what the programmer intended to create. This analysis can be done with a running program or an examination of the source code. In addition, the infrastructure that the end user has in place shapes the when and where sounds can be played.

3.1. Detailed Software Behavior

This section explores the functional use of sound in software. The sounds used in a program can be categorized by the auditory design approach, the function of the sound, and the the level of control the user has in selecting the sounds used. There are a few major types of sounds often mentioned in the ICAD literature. Auditory icons are natural-sounding representations of objects. Gaver introduced auditory icons [10] and explored their use in the SonicFinder auditory interface [11]. Earcons capture less about the object itself but more about its relation to other elements [12], such as hierarchical position. Brewster [13] determined that earcons could effectively convey information to the end user. Speech output is a common approach to auditory user interfaces. Spearcons are a non-speech audio representation of a spoken phrase [14,15]. Identifiable and small, a Spearcon is like a fingerprint of the original speech phrase. Spearcons have been shown to be useful as enhancements to menus, and as such can also be followed in a menu by the uncompressed text-to-speech phrase if the user might need access to the full text message. Soundscapes are auditory scenes. They can be natural, such as the sound in a park, or synthesized. The purpose of a soundscape can be aesthetic or informational. TAPESTREA [16] and SoundScape [17] are two examples of tools designed specifically for building soundscapes.

The sound can be an alert, ambient, interactive, or end-product. Alerts are designed to gain attention about a particular event. Emergency sirens, system warnings, and typical email notifications are all alerts. Alert design tends to match sound intensity with the importance of the event; louder sounds mean bigger warnings. Ambient information can be put into the background of the users mind. As the information changes, the user may be subtly alerted to unusual patterns [17]. Ambient audio approaches try to balance the utility of alerts with a moderation of sound distractions. Interactive audio involves hearing sounds based on the users direct activity. This mimics the role of sound with non-software objects. For example, when a person rotates their wrist while holding the handle of a maraca, the ball of the maraca visibly moves, and a sound of objects on the inside of the maraca can also be heard. The sound is a product of the interaction in a direct manner. Typical alerts don't match this approach since they are more logical, a warning that something is a problem. Interactive sound in computer systems could involve auditory menus [18] or interface widgets in general [4,5]. End-product audio is when the audio playback and recording is a system goal. For example, an MP3 player or a dictation system both rely on audio as a vital part of their use.

Audio customization effects how the user receives the designed sounds. A selection of the sounds available allows the user to self-design sounds to their own situation and preferences, much like a 'skin' for graphical interfaces. Toggles for sound types to play also allow customization. The volume level suggests different interpretations of sound. If some sounds or other programs are inappropriate, these may cause the user to change the volume, in effect dimming the auditory interface. If the volume is completely off then the auditory interface has no influence.

Property	Category
Design	Auditory icon, earcon, speech, spearcon, soundscape
Function	Alert, ambient, interactive, end-user
Custom	Customizability, volume

Table 1: Audio use properties to be measured.

The auditory design, function, and customizability broadly define a system's use of audio. Adium instant messenger for Mac OS X, for example, has alert earcons that are customizable. One measure that is perhaps missing from this study is the necessity of audio - can work in an application be easily done

without the help of sound? It is possible that future studies can look at the software identified in this study and correlate patterns of sound types used and the necessity in the system.

A list of programs used in each environment will then be inspected from an end user perspective. The programs will be evaluated to determine the system sounds. For example, a multiplayer interactive world might have interactions that use earcons and an ambient soundscape that reveals the surroundings. A finance program may have no sound at all. The list of programs will not be completely representative, and will be shaped by cost considerations. However, it can demonstrate patterns of use within and between environments. The environments themselves are subject to division, but are kept broad at this point to help reveal the appropriate separations. Another layer of environment is the user's ability to interact, i.e. the visually impaired may use computer programs differently based on accessibility issues with a graphically dominated software environment.

3.2. Source Code

Program code is the most concrete type of information in this project. The end result of this part will be to have a code grabbing apparatus, a code searching tool, and sample data gathered from searches. Since this section requires access to source code, the projects we select will all be accessible to this project with permission or through an open source license. Fortunately, large repositories of open source projects already exist (e.g., SourceForge); we intend to select one or more of these resources to act as the population of programs for our samples.

Each program selected will have its source downloaded. There are syntactic and practice differences between programming languages. However, in programs with audio, basic words such as "sound" or "audio" are probably embedded in comments, variables, or function names. Therefore, we will index every line of code and make the text searchable, most likely in a database. The database schema is simple: the most important field is a line of code from the source. There will also be fields to identify the table index of the next and previous lines; this isn't necessary for a word count but is useful in reconstructing the code to search for relevancy and dependencies. A project identifier, file identifier, and relative path distinguish all of the parts of the program. Binary resources will not be stored, so a complete rebuild and run of the code may not be possible, but a line of code can certainly be visualized in the context of the other parts of the file and project. The database will not have functional references to other parts of the code nor attempt to have a semantic understanding of what the code is doing. Due to the nature of storage, this work will exclude languages that don't have a textual representation; for example, languages that are completely graphical. Since this work focuses on current practices, it will analyze the current release of a program and avoid old versions. While old code certainly exists in long-term projects, the project itself is a representation of what programmers value as important parts of their code today.

In addition to general searches, there will be subcategories of programs. An "auditory tools" category will inform the term search parameters, since these programs certainly have the use of audio in their code.

The search tool itself will be a lightweight front end to the database. It will provide a search mechanism that returns some statistics on the term usage and access to the results in context. It will have expressive flexibility through some sort of regular expression scheme.

The results are intended to be as basic as possible; it will allow for statements such as "4 out of the 10 projects selected used the term 'midi'". As part of a sampling approach, such basic facts can greatly shape our understanding of the state of audio. For example, the use of auditory terms in a program suggests that it uses sound. The searches can reveal if the programmer is relying on a third-party library or handwritten code. This in turn suggests the level of effort a programmer put into implementing sound. Repeated attempts by several programs could suggest a role for libraries (as proposed by [4,5]). In addition, this data can be mined for comparative data on the use of other interface approaches such as GUIs and tactile displays. The categories of the projects in the source provide an opportunity to compare the programmer-perceived utility of sound in different domains. This in turn can later be compared with the actual use in the domain.

This snapshot of software code can also be utilized in the future. More than the other techniques in this project, the data from this method have many avenues of future verification and interpretation. For example, a replication of this study in five years may find changes in certain activities. If new search terms are discovered at that time, the five-year-old data can be examined using the new search. The data source is so flexible that it could be used for work completely unrelated to auditory interface research.

Literature on this subject tends to focus more on comments and their relationship to the code and bugs [such as in 19]. A simple term/word search is appropriate for our present purposes. Its simplicity also reduces the chance of error possible in automated analysis. However, if there are patterns of coding (not necessarily software design patterns as described in [20]) found in the searches, automated analysis may deliver an easier way to view auditory code sections in future studies.

Perhaps the lack of published interest in the subject has to do with the simplicity of the analysis; a searchable database of lines of code is not a particularly creative database schema or analysis technique. The difficult part of this problem is gathering the code and putting it into the system in a timely matter. The analysis can be immediate or delayed for several years, but the data have to all be gathered in a short time frame to represent software in a certain time. Since the structure of program source code depends on the language, programming culture, and whims of the developers, identifying relevant parts to insert into the project will have to be done on a project-by-project basis. This will ideally be a shared, distributed effort, involving many in ICAD and the related communities.

3.3. Infrastructure

By understanding the infrastructure, we can further understand the user's constraints in using audio with software. If the speakers are not plugged in, then no amount of effort on the programmer's end will deliver sound to the end user. This approach will ask users about their practices using computers on the hardware and operating system levels.

In the "Ethnography of Infrastructure", Star makes the case for the utility of infrastructures [21]. She describes the background nature of infrastructure and its interdependence with the user's environment. Infrastructure is often taken for granted to the point of not being an element in analysis. While Star has rather formal examples of infrastructure such as telephone books, this study will leverage the idea to analyze the system constraints on a user's activity in a system. Instead of ethnography and studying the infrastructure itself, this part will rely on user descriptions of their infrastructure environment.

The descriptions will have two components. User sketches provide a graphical interpretation of people's views on their use of technology [22,23]. They constrain the answer space in a different way than typical surveys. They also provide a point of discussion. For example, if someone draws how they use a computer and the drawing doesn't include speakers, does that mean that they aren't plugged in, or are unimportant to the user?

Follow-up questions can shape a discussion based on the participant's drawings.

The second approach involves more traditional surveys. Through multiple choice and short answer questions we will build a quantitative understanding of the equipment and its use in the user's environment. "What percentage of the time you are at your computer do you have your sound on?" Questions such as this provide general ideas about how people involve sound in computing. Combined with the technological sources, it defines what sounds make it from the software generation to the users' ears.

The visually impaired community makes particular use of audio assistive technology to successfully operate in the computing world. This section on infrastructure will pay close attention to the differences in survey results between those of the sighted and visually impaired communities. As such, this will be an element of the user demographics collected in this project. While sketching would be a valuable component in understanding any infrastructure, it may fail when working with visually impaired participants who are blind or have very low vision. However, a modeling approach or spoken or typed description may provide a way to capture the essence of a sketch.

4. CONCLUSION

The purpose of this paper is to outline our plans and to elicit responses in this field of research. The project itself is a characterization of the audio in software artifacts available to the general public. We expect that the baseline data will provide patterns found within and between the three methods of data gathering. The methods themselves were selected to provide complementary views of the situation. By focusing on the situation as it is now, we hope to provide a clearer picture of where new audio research can be directed toward.

5. REFERENCES

[1] Freedom Scientific, "Freedom Scientific - JAWS for Windows Screen Reading Software."

[2] J. Lumsden and S.A. Brewster, A survey of audio-related knowledge amongst software engineers developing human-computer interfaces, Glasgow, England: Glasgow University, 2001.

[3] J. Lumsden and S.A. Brewster, "Guidelines for audio-enhancement of graphical user interface widgets," Proceedings of HCI, London, England: 2002.

[4] S. Brewster, "A sonically enhanced interface toolkit," Proceedings of the 3rd International Conference on Auditory Display, Palo Alto, CA, U.S.: 1996.

[5] B.K. Davison and B.N. Walker, "AudioPlusWidgets: Bringing Sound to Software Widgets and Interface Components," Proceedings of the 14th International Conference on Auditory Display, Paris, France: 2008.

[6] W.E. Bijker, Of Bicycles, Bakelites, and Bulbs: Toward a Theory of Sociotechnical Change, Cambridge, Mass: MIT Press, 1995.

[7] P. du Gay, S. Hall, L. Janes, H. Mackay, and K. Negus, Doing Cultural Studies: The Story of the Sony Walkman, London: Sage, 1997.

[8] P. Dourish, "Implications for Design," Proceedings of the SIGCHI conference on Human Factors in computing systems, Montreal, Canada: ACM, 2006.

[9] Bell and P. Dourish, "Yesterday's Tomorrow: notes on ubiquitous computing's dominant vision," Ubiquitous Computing, London, England: Springer-Verlag, 2006.

[10] W.W. Gaver, "Auditory icons: Using sound in computer interfaces," Human-Computer Interaction, vol. 2, 1986, pp. 167-177.

[11] W.W. Gaver, "The SonicFinder: An interface that uses auditory icons," Human-Computer Interaction, vol. 4, 1989, pp. 67-94.

[12] M.M. Blattner, D.A. Sumikawa, and R.M. Greenberg, "Earcons and icons: Their structure and common design principles," Human-Computer Interaction, vol. 4, 1989, pp. 11-44.

[13] S.A. Brewster, P.C. Wright, and A.D.N. Edwards, "An evaluation of earcons for use in auditory human-computer interfaces," Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems, Amsterdam, The Netherlands: ACM, 1993.

[14] B.N. Walker, A. Nance, and J. Lindsay, "Spearcons: Speech-based earcons improve navigation performance in auditory menus," Proceedings of the 12th International Conference on Auditory Display, London, England: 2006, pp. 63-68.

[15] D. Palladino and B. Walker, "Efficiency of Spearcon-Enhanced Navigation of One Dimensional Electronic Menus," Proceedings of the 14th International Conference on Auditory Display, Paris, France: 2008.

[16] M. Ananya, R.C. Perry, and W. Ge, "TAPESTREA: Sound scene modeling by example," ACM SIGGRAPH 2006 Sketches, Boston, MA: ACM, 2006.

[17] B.S. Mauney and B. Walker, "Creating functional and livable soundscapes for peripheral monitoring of dynamic data," Proceedings of the International Conference on Auditory Display, Sydney, Australia: 2004.

[18] P. Yalla and B. Walker, Advanced auditory menus, 2007. GVU Technical Report. GIT-GVU-07-12.

[19] Y. Padioleau, L. Tan, and Y. Zhou, "Listening to Programmers - Taxonomies and Characteristics of Comments in Operating System Code," Proceedings of the 31st International Conference on Software Engineering, ICSE, 2009.

[20] C. Frauenberger, T. Stockman, and M.L. Bourguet, "Pattern design in the context space: A methodological framework for auditory display design," Proceedings of the 13th International Conference on Auditory Display, Montreal, Canada: 2007.

[21] S.L. Star, "The Ethnography of Infrastructure," American Behavioral Scientist, vol. 43, Nov. 1999, pp. 377-391.

[22] M. Tohidi, W. Buxton, R. Baecker, and A. Sellen, "User sketch: A quick, inexpensive, and effective way to elicit more reflective user feedback," NORDICHI, 2006.

[23] E.S. Poole, M. Chetty, R.E. Grinter, and W.K. Edwards, "More than meets the eye: Transforming the user experience of home network management," Designing Interactive Systems, 2008.