Investigation of multiple visualisation techniques and dynamic queries in conjunction with direct sonification to support the browsing of audio resources

Thesis

Master of Science in Computer Science

Eoin Brazil

Interaction Design Centre

Dept. of Computer Science & Information Systems

University of Limerick

Limerick, Ireland

Supervisor: Mikael Fernström M.Sc.

This Thesis is presented as fulfilment of the requirements for a Master of Science in Computer Science. It is entirely my own work and has not been submitted to any other university or higher education institution, or for any other academic award in this University. Where use has been made of the work of other people, it has been fully acknowledged and referenced.

Submitted to the University of Limerick, July 2003.

Abstract

In this thesis, we have developed a prototype system for the browsing of audio resources and performed an initial evaluation of this system. The main contributions of this thesis are dynamic queries and multiple visualisation techniques in conjunction with direct sonification. Dynamic queries are queries that provide immediate feedback while maintaining consistency between the queries themselves and the graphical/auditory display. The multiple visualisation techniques are used to present different representations of collections of audio resources. Exploring the multiple representations allows for the discovery of new relationships within the audio collection. Existing techniques and mechanisms were examined from the areas of visual and auditory browsing; the conclusions drawn from this examination informed our design choices for the prototype system. The exploratory investigation of the prototype system focused on the specific domain of sound design. A series of tasks in the domain gauges the users subjective experience and investigates the use of our system within a real-world situation.

Dedication

To my parents for their support and encouragement and convincing me that I can do anything I set my mind to.

Acknowledgements

Friendly counsel cuts off many foes.

- William Shakespeare (King Henry VI, Act III, Scene 1)

First, I would like to acknowledge the support and advice offered by Mikael Fernström of UL without whose initial enthusiasm the research would not have been undertaken. Without his tireless and patient mentoring, it might not have been completed. Prof. Liam Bannon at the Interaction Design Centre provided the advice and guidance in ensuring that this thesis was completed. I still take too many words to say things, but now I'm more aware of the value of brevity. Mark Marshall, Bruce Richardson, Tony Hall and Krispin Leydon tackled the equally important practical side of things by keeping my horizons broadened by their work. In addition, special thanks are due to Colm McGettrick who not only provided technical support but also a different viewpoint on the work. Thanks to Laura Ottaviani for her assistance in running the experiment and to all the subjects who volunteered their time to participate in my experiments. A special thanks to Anne Murphy who was forever there to provide help and advice. It has been a distinct pleasure and enlightening experience to interact with Mikael, Liam, graduate students and researchers in the IDC at UL who are blazing trails for the future of sound and HCI.

The European Community under the 'Information Society Technologies' programme sponsored this research under IST project number IST-2000-25287. Special thanks to those involved in the Sounding Object Project (SOb) especially Davide Rocchesso, Nicola Bernardini and Roberto Bresin.

Table of Contents

Tables

Figures

Chapter 1

Introduction

1 Introduction

Architects of large projects use many types of plans, drawings and models that offer various views of the new structure. Each view has advantages and disadvantages, but in combination they offer an unparalleled composite view of the new structure. The ability to examine an object or collection from multiple different perspectives offers new possibilities for exploration and discovery of previously hidden relationships within the data. This idea to provide multiple different visualisations of audio collection has become one half of the focus for this thesis and it allows many different perspectives on sound collections. The second half of the focus of this thesis is to provide dynamic queries, which allow for results to be immediately returned to the user while maintaining a consistent representation of the collection. The summary of requirements for this research is presented in section 1.3.

Our goal is to offer an interface which engages the user and is based on the Direct Manipulation paradigm of Shneiderman (1992a). This type of interface offers a directness of action and a feeling of directness. The users have a feeling of control over the objects in the task domain. This results in a "Direct Engagement" (Hutchins et al., 1986) of the user who feels that they are operating directly upon the objects, where the objects upon which the actions are performed by users are the very same as those from which the feedback is received by the user. Hutchins et al (1986) p123 state that a "*properly developed Direct Manipulation interface should appear to the user as if it is the task that is being executed directly. The Computer system and its interface will be more or less invisible*."

Previous research in this area includes the "Novel Multimedia Browser Interface Mechanisms" project and the later research into *The Sonic Browser* research project (Fernström and Bannon, 1997b). The result has been a substantial foundation of work developed on browsing and sonification (Fernström and McNamara, 1998). The earlier

work by Fernström on the BROWSE and on the Sonic Browser systems offers the following functionality:

- "star-field" display for representation of the data set (see Section 2.2.5)
- tight coupling between the visual and auditory representation (see Section 3.7)
- multiple stream stereo-spatialised audio (see Section 2.4.5)

In our prototype system we implement these areas of functionality and add the following areas of functionality:

- multiple visualisations for alternative representations of the data set
- dynamic queries (see Section 2.2.5)

By exploiting novel graphical interface mechanisms designed for large-scale datasets, combined with dynamic query mechanisms such as dynamic sliders and multiple visualisation techniques, we aim to investigate the use of our prototype system within a real world situation and to examine the user's experience of our system.

The development of audio browsing systems has been influenced by the existing interface metaphor of the tape recorder / player, with play, stop, fast forward, rewind, pause and possible record buttons / functionality. We propose that this may not be the most effective interface metaphor as due to its conceptual design, only a single audio resource can be played at any one time. Another problem with the development of audio browsing systems is that specific elements concerned with the use of sound in interfaces are often considered either an academic topic or esoteric lore by commercial software developers. Many modern applications demonstrate this difficulty, an example being the single sound playback of the most widely used browser, Microsoft's (MS[1]) "Windows Explorer" (Fernström and Brazil, 2001). In the instance of the "Windows Explorer", the difficulty while once related to hardware difficulties in playing more than one sound simultaneously is now related to the original conceptual design for the interface, which was originally designed with regard to the hardware difficulties. In browsing one sound at

[1] MS stands for the Microsoft Corporation. In this thesis, names of computer programs, etc., which are preceded by MS are registered trademarks of the Microsoft Corporation.

a time a limit is placed on how we normally perceive sounds. We are subjected constantly to a barrage of sounds within the world and often many sounds simultaneously (Cherry, 1953, Arons, 1992). A good example of the multiple simultaneous sounds can be heard in the different layers of audio in a movie soundtrack - including music, dialogue, ambient sounds, footsteps, dripping taps / pipes, gun-shots (Sonnenschein, 2001). Our research aims to exploit human abilities for both auditory and visual processing and as such our prototype systems offers the ability for the playback of multiple sounds concurrently. This could allow for a more engaging presentation of the information and may also offer other benefits such as the discovery of new relationships between sounds.

Prior to designing any new style of interface for presenting a collection, audio or otherwise, we review the previous techniques used to address the problem, in this case the various auditory and visual techniques. This review is helpful in outlining the existing techniques and highlighting any pitfalls or known problems that have been documented in the research literature. This was the starting point of our research as it would not only suggest techniques for inclusion in our prototype system, but it would also helps provide a basis for our examination of audio browsing systems. A review of the existing audio browsing systems was the next area investigated in this research and this review is presented in Chapter 3. The lessons learned from examining the previous systems and the existing *Sonic Browser* application provide a foundation upon which to build our prototype system. The prototype system also implements dynamic queries and multiple visualisation techniques whose design have been influenced by the knowledge gained from this review.

1.1 Browsing Audio Collections

The amount of audio information and resources available on personal computers and via the Internet has grown exponentially over the past decade. This is referred to as the data availability paradox (Woods et al., 1999): as more and more data is available, but our ability to experience what is available has not increased. When considering the number of these resources, it is evident that there must be ways to provide richer and more interesting systems that allow us to perceive more as the amount of data increases (Hutchins et al., 1986). Designing systems which provide engaging and more intuitive interfaces was the initial idea upon which the early foundations of this work are based. The systems we are concerned with relate to browsing sound collections.

Browsing is defined by the Oxford Reference Dictionary as:

- to casually read or inspect items on display.
- (of animals) to crop or feed on leaves or young shoots.

Researchers have used these two definitions, but it is primarily the first definition that people associate with the word "*browsing"*. The interpretation of the first definition which will be used in this research is that given by Marchionini & Shneiderman (1988) p71 who interpreted browsing as "an exploratory, information seeking strategy that depends upon serendipity … especially appropriate for ill-defined problems and for exploring new task domains". Marchionini defines information seeking as "a process in which humans purposefully engage to change their state of knowledge" (1995) p5.

The two areas highlighted by Marchionini are "ill-defined problems" and "exploring new task domains". This is the case when sound designers are browsing for sounds in a collection in order to create a new sound. Imagine an example with Bob, a sound designer who is building gunshot sounds for an action scene in a movie. Bob needs to find several sounds to create a new sound, which will be heard, as realistic. He looks at a recording of a gunshot from three points: the point from where it was shot, a point half way to the target and at the target itself. Another example of a sound designer is Alice

who is creating a sound track for a new nature program on lions by David Attenbourgh. The problem for Alice is that the video was shot using a telescopic lens. The distance resulted in a poor sound recording so Alice must now create an entire sound track to replace it. Various sounds are needed from paws through soft grass, to lion roars, to a number of bird songs. The bird songs are included as they add realism to the sound track when layered with the other sounds. This particular setting forms the basis of our investigation which we conducted using our prototype system. The work of sound designers and our scenario is described in greater detail in Chapter 5.

By providing a system with different visualisations of audio collections and dynamic queries, we investigate the use of our prototype system within a real world situation. One of the foundations of Fernström's work in direct sonification (1998, 1998) is that "Humans have an outstanding ability to recognise similarities in this domain, which suggests that a good solution should make use of our auditory abilities" that is, to get a tool that is engaging for the users to work with and that supports both the user's visual and auditory senses.

The main difference between browsing and directed searching is that, with browsing, a user may not have any specific sound in mind but may wish to explore the contents of the collection, while with directed searching the user is looking for a particular sound or type of sound. Directed search strategies are dependant on planning, on the recall of the particular query, as well as an iterative refinement of the query from the examination of the query results. Browsing strategies are heuristic and opportunistic, and depend on recognition of relevant information. Directed searching is similar to a turn based game or human conversation where one participant waits for the actions of another before continuing. This differs from browsing strategies, which are more interactive, real-time exchanges between the participant and the information space. An example of this difference would be where a user was searching for a the sound files containing the engine noise of a Ferrari 1989 sports car starting up and driving away, whereas with browsing a user looking for any sounds which contain cars or car sounds, and which may be part of a large auditory soundscape such as a busy urban street recording. This is an

important difference and highlights the two different audio interface paradigms of browsing and directed searching. The different goals lead to different techniques to meet the user's requirements. The two types of approaches can operate in conjunction, but it is important to realise their foundations, as these influence their respective strategies to finding or browsing the desired information.

This research provides a tool that allows users to use both their visual and auditory abilities to recognise sound similarities when browsing for audio resources. In this thesis, a prototype system to allow for such browsing and to support it with dynamic queries and multiple visualisation techniques is presented. Our approach is to support the user's browsing with multiple stream audio and various techniques such as direct manipulation, dynamic queries and multiple visualisations in an effort to provide a usable tool.

1.2 Overview of Previous Research

Earlier work in the area of combined graphical and audio interfaces for the browsing of audio resources has concentrated on standalone applications run, from a single workstation, using 2 dimensional or 2.5 dimensional graphical user interface's (GUI's). The main issues for applications in this area, as well as reviews of several applications, will be covered in greater detail in section 3.6, where comparable systems to the Sonic Browser are examined. The comparable systems examined include the Microsoft Explorer, the Ninja Jukebox (Welsh et al., 1999), the Muscle Fish system (Wold et al., 1996) and the Marsyas (Tzanetakis and Cook, 2000c) and its derivatives (Tzanetakis and Cook, 2000a, Tzanetakis and Cook, 2001, Tzanetakis and Cook, 2000b). By studying these applications we can gain an insight into aspects of the problems facing audio browsing interfaces and find techniques that have already proven effective. We have also examined the previous work on the Sonic Browser.

We have investigated different interface mechanisms, both graphical and auditory, to determine which are effective in either specific domains or for general use. Applications that have relevant interface mechanisms reduce the workload of the user and make the tool's use more engaging.

Results of several earlier studies in the auditory domain have influenced this research. For recognition of sounds, it is often sufficient to hear only 500 milliseconds to 2 sec of the characteristic or significant part of a sound file (Warren, 1999). Humans are remarkably good at genre classification as investigated by Perrot and Gjerdigen (1999) where it is shown that humans can accurately predict a musical genre based on 250 milliseconds of audio. These recognition results were the basis of auditory hypothesis used in this thesis along with research on the identification of sounds (Ballas, 1993) that suggests that these classifications could assist in the browsing of sounds. However, these classifications are affected by the variance of individual perceptions and, as such, any form of classification of audio is highly subjective.

In reviewing the previous literature there was one particular family of techniques that were particularly well suited to the presentation of a collection. These techniques come from the visual browsing literature and are the various *focus and context* (F+C) techniques (Lamping et al., 1995). Lamping defines F+C techniques as those "in which detailed views of particular parts of an information set are blended in some way with a view of the overall structure of the set" p401. These techniques are used to enhance a particular set of elements within the scene while still keeping all the other elements present as context. They provide a user-controlled focus point for indicating which part of the data is to be shown in detail. These techniques are a solution to the space problem in information visualization, and allow many more objects to be displayed than would be possible in an undistorted view. They also provide a way for addressing the space problem of collections and provide an area for investigation when applied to the browsing of audio resources. A number of these techniques employ spatial distortion-oriented methods (Leung and Apperley, 1994), i.e., where the parts or elements of interest are enlarged but the other parts or elements are scaled or shrunk to fit the remaining display space. This type of visual technique as well as other visual and auditory techniques are examined in further detail in Chapter 2.

We have used these *focus and context* techniques, combined with auditory display mechanisms in order to create novel interfaces for browsing auditory resources. The Audio Retrieval Browser (ARB) prototype system (Brazil et al., 2002a) was designed as the first exploratory application in this research, combining the MARSYAS (Tzanetakis and Cook, 2000c) software framework with the Sonic Browser (Fernström and McNamara, 1998). The goal was to provide an interactive application with sound classification using the best features of the two previous applications. The Audio Retrieval Browser was the first prototype of this application. The ARB uses a flexible, distributed, graphical user interface with multiple visualization components, combined with automatic classification of musical sounds into musical genres to provide novel ways of browsing large audio collections. The work in this thesis on the Sonic Browser builds upon the experience gained from the implementation of the ARB, while seeking to further refine audio browsing interfaces by examining the success of different

visualisation mechanisms in an exploratory empirical investigation. The empirical study was used to investigate the use of our prototype system within a real world situation.

This thesis examines the mechanisms of dynamic queries and multiple visualisations for audio collections. The research on our prototype system does not include any classification of sound, as this is a difficult and complex topic beyond the scope of the current work.

1.3 Empirical Study

The scenario used in our exploratory empirical study was undertaken as part of a larger project called the Sounding Object (SOb). The Sounding Object Project has pioneered several recent attempts to couple physical simulations to efficient sound synthesis techniques. The European Commission funded this project to study new auditory interfaces for the Disappearing Computer initiative[2]. The SOb project concentrated on four areas: sound model design on perception to find the ecologically relevant auditory phenomena with psychophysics for help in organizing access to their parameters, investigation into whether physics-based models can be 'cartoonified' to increase both computational efficiency and perceptual contrast; investigation into physics-related variables of sound models which can be varied in time and organized in patterns to convey expressive information and the construction of interactive systems around sound and control models. The reports, articles and software from the project can be found at the SOb website[3].

A sound model is a synthetic caricature of a sound, where the salient features of the sound have been identified and can be parametrically controlled. The sound models use complex responsive behaviours that are designed to be expressive in an ecological sense, rather than static sound file representations of a sound. The sound models use various simplifications when representing the sounds, preserving certain acoustic aspects, while eliminating other less prominent perceptual aspects. Using a "cartoonified" or simplified representation of a sound, a sound model returns a dynamic representation of that sound whose computational costs are lower.

The development of parametric models (Sound Objects) lead to the development of our prototype system for testing and validating the sounds produced by the sound models, as well as for the management of collection of these sound models (Brazil et al., 2002b).

---

[2] http://www.disappearing-computer.net

[3] http://www.soundobject.org

Our system was developed as a tool for accessing sounds or collections of sounds, using sound stereo spatialization and F+C visualisation techniques. The management and use of a large collection of these sound models is our scenario.

The empirical study was performed in the area of sound design and how sound designers manage and browse sound collections for the production of sound effects. Sound designers are often called upon to create sound effects that are composites of the various sound resources available. The currently available sound editors do not allow for easy browsing of large sound collections to quickly hear sounds and to determine if they are relevant for the new composite sound. Sonic browsing helps users compose new sounds by letting them interactively access large sets of sounds; that is, to pick, group, and choose what sounds might work together as a new sound. The possibility of addressing some portions of the issue of new sound composition motivated this scenario. The empirical study investigates the use of our prototype system in real-world situations encountered by sound designers.

1.4 Research Objectives

The focus of this research has been the investigation and review of the area of audio browsing – the systems, techniques and mechanisms and the development and initial evaluation of a prototype distributed graphical user interface with multiple visualization components and dynamic query mechanisms for the browsing of audio resources and examining the user's subjective experience. The research is based on previous research carried out in the fields of sound browsing, sound classification, audio retrieval, audio interfaces, image retrieval / browsing, human computer interaction and information visualisation. All these areas contribute to the retrieval and browsing of resources, whether graphical or audio, as well as providing the principles, tools and techniques necessary to design a new interface. In creating this prototype system, we need to provide multiple visual representations and consistent, interactive, dynamic querying to allow for the investigation of relationships within the dataset, and with our exploratory study we seek to discover the user‘s subjective experience of our prototype system.

1.5 Contribution

The work includes an investigation of the existing techniques and systems for browsing, the development of a new prototype system and the initial evaluation of the new system.

The investigation focused on reviewing techniques and systems for browsing from earlier research literature. The development of the prototype consisted of creating a system for browsing audio resources using several visualisation techniques and dynamic queries. The visual and auditory interfaces provide multiple views of sound collections. The querying mechanisms allow for the focusing on areas and objects of interest.

The initial evaluation of the audiovisual interface sought to discover the user‘s subjective experience of our prototype system, and investigate the use of our prototype system in a real world situation. The evaluation examines visualisation mechanisms in conjunction with dynamic queries for specific use scenarios. We also provide an application framework that can be used for further investigation and exploration into the area of sound browsing, either directly, as an application, or indirectly as a basis upon which to build further interfaces using the underlying foundations of our prototype system. The findings of this initial study are refined and generalised to provide suggestions for interface mechanisms for sound collections.

1.6 Organisation of Thesis

This thesis presents a prototype system for the browsing of large datasets of sounds using dynamic queries and multiple visualisation techniques. It investigates the user's acceptance of our prototype system and the suitability of the system and its mechanisms. Chapter 2 presents a review of the literature and techniques in the areas of sonification and browsing. A short description of the major areas within sonification is presented. This review divides browsing into two areas, the browsing of graphical resources and the browsing of sound resources. It highlights several auditory and visual browsing techniques, focusing on the mechanisms which use the *focus + context* presentation of a collection or dataset, as well as presenting some graphical user interface components or widgets. Chapter 3 consists of a review and an examination of various related systems used for the browsing of sound resources. It proceeds with an overview of the previous versions of the Sonic Browser system with the issues of relevance to this research highlighted. It then examines the functionality of the existing applications and possible areas of improvements within these applications, based on the previous systems and applied to our new prototype system. Chapter 4 discusses the implementation, and provides a brief overview of the design we used to create our prototype system. Chapter 5 presents the empirical study and the history behind its adoption for our research. It then follows with an overview of various procedures and techniques for the testing of use scenarios and gives information regarding the participants, the experimental design, the experimental set-up and the procedure used for conducting the empirical study. Chapter 6 further expands upon this empirical study, and displays the study's results with a discussion before offering some conclusion that can be derived from the results of the study. Chapter 7 provides a summary of the work and the contributions made by this thesis. It then offers general conclusions and suggests some future directions for research into the area of sonic browsing.

1.7 Chapter Summary

In this chapter, we have presented an introduction and outlined a definition for sonic browsing. We have also outlined the contribution and the organisation of this thesis. A brief overview of our prototype system was discussed in the preceding section.

In the next chapter, we review the existing research literature on sonification and browsing.

Chapter 2

Review of Research Literature in Sonification & Browsing

2 Review of Research Literature in Sonification & Browsing

This chapter presents a review of research literature in the areas of Sonification and Browsing. The earlier research on sonification is presented as it relates to our research and various topics including auditory icons, auditory scene analysis and the "cocktail party" effect as they provide the theoretical underpinning. The earlier research on browsing is divided into two distinct categories; visual and auditory. These categories review several techniques for browsing and their particular advantages and disadvantages. The chapter concludes with a review of suitable techniques for browsing audio collections and a selection of the most suitable of these techniques for inclusion in our prototype.

Before reviewing earlier research we must first define the type of resource collection we are interested in. This research focuses on audio resources collections. Audio resources can be classified into two major categories, speech and non-speech sounds. The latter category can be further divided into those that deal with everyday sounds and those that deal with music (Melih and Gonzalez, 1998). These categories have been influenced from previous work by Macaulay et al (1998), by Gaver (1993c) and by Wold et al (1996). Macaulay et al classified elements of a soundscape based on sound type, information category and acoustical information. Gaver's hierarchical descriptions were used when the sound type was pre-classified as an everyday sound or an abstract sound. Wold et al proposed the classification of sound using simile, acoustical/perceptual features, subjective features and onomatopoeia. These classification schemes influenced our prototype's mapping of sound properties, which we discuss in Chapter 4. The importance of distinguishing the various properties under which sound may be classified helps in determining the suitable mechanisms and techniques from the review of the earlier research literature.

2.1 Direct Sonification

There are several definitions of sonification. An earlier definition was "*the transformation of data by a sound generator to the classical dimensions of sound such as frequency, amplitude, and duration for monitoring and comprehension*" (Scaletti, 1994) p224. Sonification is also defined as "*the transformation of data relations into perceived relations in an acoustic signal for the purposes of facilitating communication and interpretation*" (Kramer et al., 1999) p3 or defined as "*the use of non-speech audio to convey information such as that used in the interpretation of scientific results*" (Walker, 2000) p17. We use the definition of sonification as a coupling between a data set and a set of acoustic signals aimed at assisting the comprehension of the data set. Direct sonification refers either to the immediate aural feedback to user actions or to the actual sonification of an audio resource. This can lead to confusion at times so, it is important to be aware of the dual meaning of the term. In the following paragraphs we introduce four important concepts that have contributed to the theoretical underpinning of the sonification aspects of this thesis. The first of these areas is that of auditory icons.

Auditory icons are everyday sounds (environmental sounds) designed to convey information from computer events (Gaver, 1994). The Sounding Object Project (SOb) has pioneered several recent attempts to couple physical simulations to efficient sound synthesis techniques. The development of parametric models (Sound Objects) lead to the use of the Sonic Browser for testing and validating the sound models (Brazil et al., 2002b). A sound model is a synthetic caricature or "cartoonified" representation of a sound, where the salient features of the sound has been identified and can be parametrically synthesised and in the case of the SOb project are based on everyday sounds. These synthetic caricatures take the most perceptually relevant features of a sound and amplify these features in the new "cartoonified" version of the sound. The relationship between auditory icons and sound models is that auditory icons provided the initial foundation of the work on sound models for the SOb project. These sound models form a part of the sound dataset used in our scenario. The sound models use complex responsive behaviours, which are designed to be expressive in an ecological sense, rather than fixed static sound files and are capable of real time change. The models are based on everyday sounds, but realistic rendering is not necessarily the perfect goal. The models

use various simplifications, preserving certain acoustic aspects, while eliminating other aspects, which are perceptually less important (Rath, 2002).

Several applications have used auditory icons in the interface as a mechanism for returning feedback to the user based on their actions and also about system events. The *SonicFinder* (Gaver, 1989) was the first interface to use auditory icons to give user's feedback concerning events and interface element selection. These included events such as for selecting, dragging and copying files, opening and closing folders, selecting scrolling and resizing windows, and dropping files into and emptying the trash can. Other applications in this area included the *ARKola* (Gaver et al., 1991) a complex sound simulation of a bottling factory and *SharedARK* (Gaver and Smith, 1990) a virtual physics laboratory, *Sharemon* (Cohen, 1994), *Mercator* (Mynatt and Edwards, 1992) Auditory icons are environmental sounds that have a semantic link with the object they represent and have been shown to be an effective form of managing attention in complex activities (Brock et al., 2002). An example application which uses auditory icons and musical songs to manage attention, is the WebMelody (Barra et al., 2002) which uses auditory icons to present the current load on a web server by sonifying the load using auditory icons and continually playing this sonification. A Webmaster can listen to this sonification in the background and can be prompted to attend to the web server when the appropriate sound combinations play signifying the need for the intervention of the Webmaster. The semantic link of auditory icons is not arbitrary, but based on an analogy with the everyday environment. A well-chosen selection of environmental sounds can be easy for a user to understand and an example would be the wind or wave patterns suggested by Conversy for process monitoring (1998). The previous applications all have one core similarity, which is the use of auditory icons at the widget level. Widgets are graphical or auditory interface components such as buttons and menus. Our research focuses on the use of auditory icons at the content level of the sound rather than at the widget level but the foundations of both uses are the same. Auditory icon's can be described and comprehended in an ecological sense (Gaver, 1993a, Gaver, 1993c). Gaver has previously described several algorithms for the synthesis of auditory icons (1993b). Mynatt has described a design strategy for the identification of suitable auditory cues for

use in auditory icons (1994). The second concept that has contributed to our thesis from earlier sonification research is that of auditory scene analysis.

The research into the area of auditory scene analysis has provided findings in the area of sound perception, scene perception and perceptual grouping. Auditory scene analysis is the perceptual grouping of auditory stimuli into auditory streams, each of which corresponds to specific auditory source. A series of footsteps, for example, each represent individual sounds, yet are usually experienced as a single source. Streams are a way of putting sensory information together. The world is a rich sensory environment with people being exposed to hundreds of different sounds simultaneously and still being able to sort out the important parts of this environment or "*auditory scene"*. A great variety of research relating to perceptual grouping of auditory stimuli into streams has been done, and summarised, by Bregman (1990).

The perceptual grouping can be described by heuristics based upon the *Gestalt* principles of psychology (Koffka, 1935). In the case of music, there are several different factors that influence the human ability to differentiate or select between overlapping auditory elements. *Auditory streams* can be supported in their formation by the use of instrumental sounds, timbre, envelope, tonal range and spatial cues. Musical sounds can also assist in the formation of auditory streams, as music has its own inherent syntactic and semantic properties (Serafine, 1988). The perceptual grouping of auditory elements into streams influenced our decision to use playback of multiple sounds. The final two concept which contributed to the theoretical underpinning of this thesis are the concepts of an *aura* and of the "cocktail party" effect.

An *aura* (Benford and Greenhalgh, 1997) is a metaphor for a user controllable function that can make this problem visible to the user and allow them to take action. An aura in the context of both the Sonic Browser and our prototype is a function that indicates the user's range of perception in the domain. The aura is the receiver of information in the domain. In our research, the aura metaphor is used in the interface to allow for browsing of audio by playing all the icons under a circular cursor, which we use in our system to represent the aura as shown in Figure 1. The cursor is controlled by the user and may have it size increased or decreased, while any sounds under it are played. It is the cursor, which indicates the user's range of perception and allows for the zooming in

and out of the perception range by increasing or decreasing the aura's size. Another human feature in this area is the "cocktail party" effect. This is the human ability to switch attention between different sounds at will (Cherry, 1953, Cherry and Taylor, 1954, Arons, 1992). Individual ability to differentiate between multiple sound sources can vary. The "cocktail party" effect was the basis for our decision to make the playback of multiple sounds simultaneous. We propose that using an aura and the "cocktail party" effect creates a more engaging experience whilst increasing the number of sounds, which can be browsed.

The four areas of auditory icons, auditory scene analysis, *aura* and "cocktail party" effect influenced our design from an early stage and the influences from these concepts can be easily identified in the auditory aspects of our prototype system.

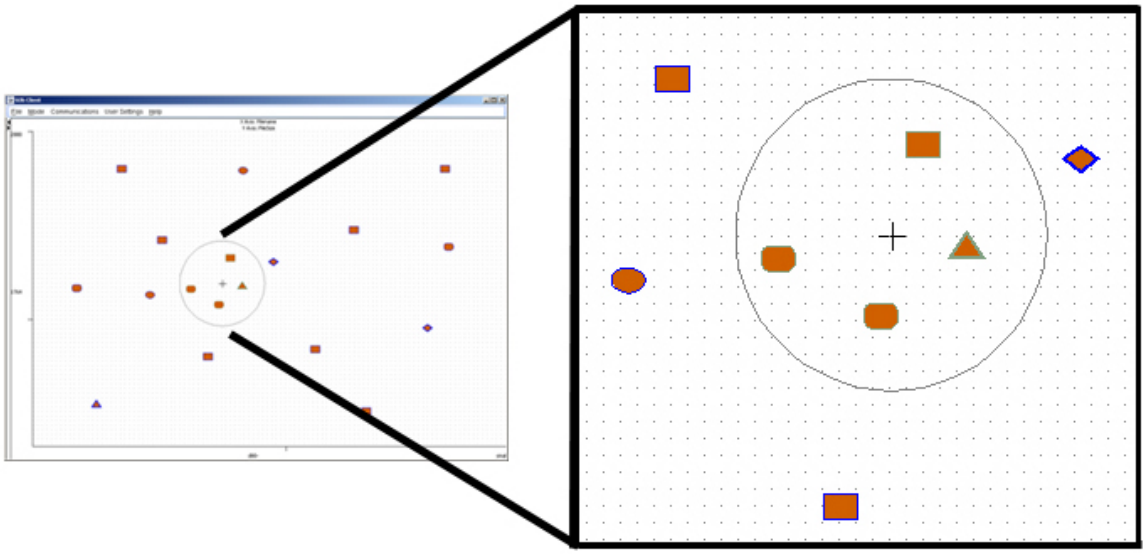


Figure 1: Our prototype system's aura represented by the circular cursor under which all objects (sounds) are played

2.2 Visual Browsing

Computers are increasingly used to store media such as photographs and music files. This has been spurred on by the now, commonplace availability of digital photographic still and video cameras as well as MP3 players (Knudsen, 1999) amongst other commercial products to the home user. The increasing ubiquity of digital collections has lead to various avenues of research focused on the searching and browsing of collections. In this section, we focus on one of the avenues of research, that of visual browsing. Visualisation has used techniques to produce several browsing systems from which we have used elements. There are numerous different applications, techniques and solutions available for the visual browsing of data sets.

The direct manipulation paradigm mentioned section 1.1 which is the main interface metaphor in our design and deserves some elaboration prior to our investigation of existing techniques as allows for a greater clarity when reviewing the possible techniques and later systems for inclusion into our prototype system. The direct manipulation paradigm is an interface design concept in which the user manipulates or selects objects on a screen with a pointing device rather than typing in syntactically structured commands. Direct manipulation is based on what Hutchins calls *model world interaction* (1989). The interface provides an environment and functionality or tools that the user applies directly solve a problem. More concisely, it is the user acting through a direct manipulation interface, rather than conversing with it. This removed need for users to decompose tasks into syntactically complex sequences. The result is that each command is a comprehensible action in the problem domain where the result of the command is immediately visible. The closeness of the command action to the problem domain reduces user problem-solving load and stress. The most common explanation of this concept is in terms of Hutchins's (1989) concepts of the "gulf of evaluation" (the user must interpret the display) and the "gulf of execution" (the user must determine how to act on the system). Direct manipulation interfaces are thought to reduce these two "gulfs", meaning that the user can more easily understand the system state revealed on the display and more easily figure out how to act on the system to achieve the desired result.

The *focus and context* techniques mentioned in section 1.2 are the techniques of interest to this research as they provide a way for addressing the space problem of

collections and provide an area for investigation when applied to the browsing of audio resources. The common thread to these types of techniques is that they allow the user to focus on objects of interest while maintaining a context through a peripheral view. In the following pages several of these techniques will be explored. The *Fisheye view* (Furnas, 1986), the *Hyperbolic Tree* (Lamping et al., 1995), the *H3 algorithm* (Munzner, 1997) are focus and content techniques whose strength is in the visual presentation of data sets, with an emphasis on hierarchical organisation. *Tree maps* (Asahi et al., 1995) and *Starfield displays* (Ahlberg and Shneiderman, 1993b, Jog and Shneiderman, 1994) are techniques that do not emphasis hierarchical organisation. *Tree maps* are a visualisation technique that displays the relative size of the objects in the dataset and may also use the colour of the objects to represent their properties. *Starfield displays* are used to display as many object properties as possibly through graphical encoding of location, colour and shape.

We use two other techniques that, while often employed in focus and context techniques, are separate techniques. The first is that of *zoomable* interfaces such as *Pad/Pad++* (Bederson and Hollan, 1994), *Jazz* (Bederson and McAlister, 1999) or *Zoomworld* (Raskin, 2000b). *Zoomable* interfaces are also referred to as the *Zooming Interface Paradigm* (ZIP). The second technique is that of *clustering* (Herman et al., 2000) which is demonstrated by the *TimbreSpace Browser* (Tzanetakis and Cook, 2000c) and the *H-BLOB (Hierarchical BLOB) algorithm* (Sprenger et al., 2000).

Photograph browsing can be a particularly serendipitous activity and as such is quite similar in nature to sonic browsing. The research into the area of image retrieval has been spurred on by the availability of digital photographs. Thumbnail (miniature) pictures and various spatial techniques have shown improvements in browsing and retrieval of images (Rodden et al., 2001, Jose et al., 1998, Kang and Shneiderman, 2000). We have found similarities in many research problems between the two areas of image browsing/retrieval and sound browsing. A good example of an image browsing application is the *PhotoMesa* (Bederson, 2001), which uses several of the techniques covered in this chapter. The area of image browsing is a closely related area of research and has provided many techniques that can be directly applied. These techniques vary from sonic thumbnails to modified visualisation techniques such as the *Quantum*

Treemaps and *Bubblemaps* both of which are used in the *PhotoMesa* application (Bederson, 2001) shown in Figure 2. *Quantum Treemaps* are a modification of the existing *Treemap* algorithm to overcome the arbitrary aspects ratios of the photographs. This is a critical requirement for photographs, as their aspect ratios must be maintained to allow for the photograph to retain it's meaning. The *Bubblemap* algorithm is used to generate non-rectangular groups of pictures (in PhotoMesa) to create an ordered space-filling layout. It differs from the *Treemap* in that it does not subdivide rectangles (or pictures in PhotoMesa) rather it uses a pixel-based bucket fill algorithm. The resulting shape is arbitrary shaped, not rectangular shaped. Combining these techniques into a Zoomable User Interface (ZUI) with a *hierarchy tree* (Directory tree in this case) completes the *PhotoMesa*[4].

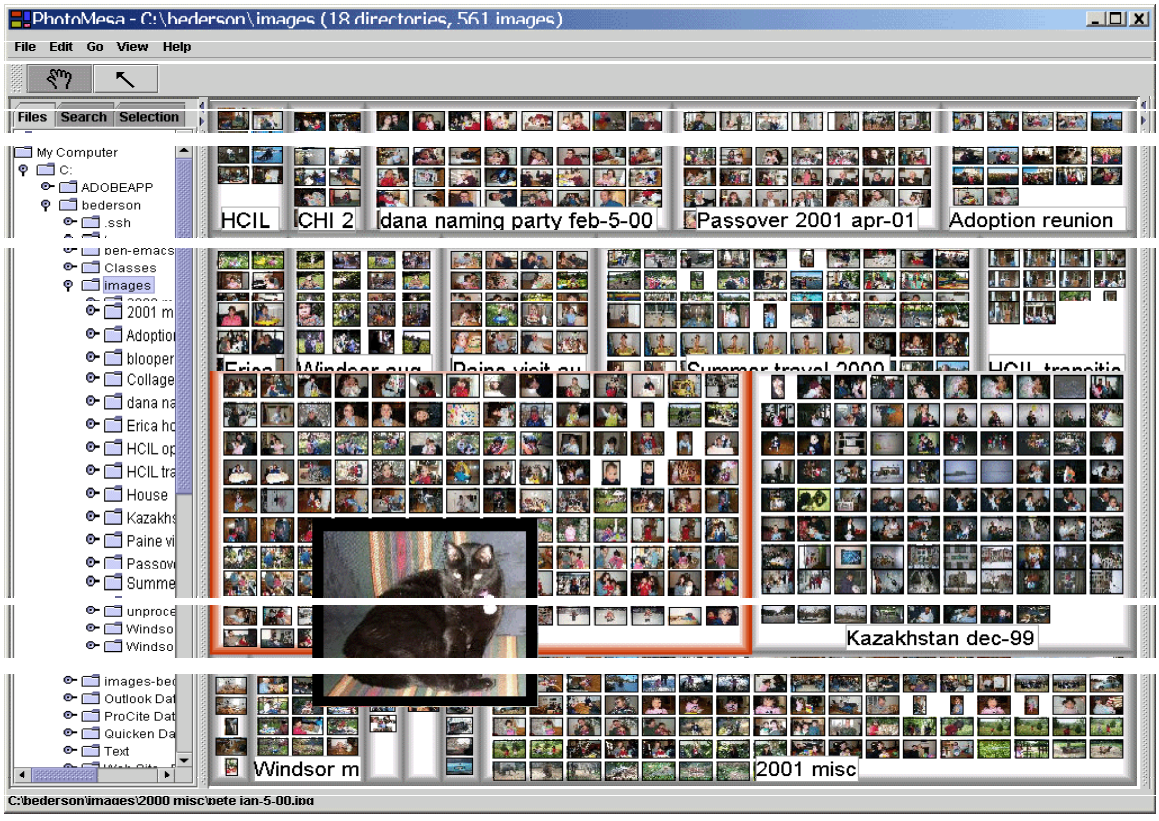


Figure 2: An Image Browser "PhotoMesa", (Source: HCIL, Uni. Maryland)

[4] The application was written using *Java 2* and is available on the Internet at http://www.cs.imd.edu/hcil/photomesa.

In the following subsections, we'll discuss the methods and techniques mentioned and highlight their particular advantages and disadvantages from the perspective of inclusion as suitable methods or techniques in our prototype system. The visual browsing techniques form one half of our review. The second half of our review will focus on auditory browsing, reviewed in the next section. The first technique we'll discuss is the one from the focus and content family of techniques, the Fisheye View.

2.2.1 Fisheye View

The Fisheye View was originally proposed by Furnas (1986) as a technique for visualising hierarchical data. As sound collections are usually divided into directories within a file system there is a readily available hierarchical classification with this file system layout which can be used for any technique which requires hierarchical data. This technique's core foundation is the method of *thresholding*. Each element of the hierarchical dataset is allocated a number based on its relevance within the dataset. A second number is allocated based on the dataset element selected and the point of focus of the dataset. This generates a threshold value, which compared with a function of these two numbers determines whether the element is to be presented or suppressed. The *degree of interest* (DOI) function is used to determine for every point in the dataset the level of interest for the point with respect to the current point of focus. Combining the prior degree of interest (DOI) values with respect to the current point at the focus and the threshold value results in the Fisheye View. There has been some confusion in the terminology as several applications use the term "Fisheye View" for different domains and also different forms. This is expanded upon by Leung et al (1994).

Further work by Sarkar and Brown (1992) on the original concept resulted in a graphical representation. A comparison of a graph using the Fisheye view and a standard linear layout is shown in Figure 3. A Fisheye view of a graph shows the area of interest quite large and with detail, and the remainder of the graph is shown in successively smaller size and with less detail. This allows the user to retain a complete overview of the dataset but with only a few objects in focus. This can be seen in Figure 3 where a normal cartographic map is presented and compared with the same map displayed as a Fisheye view supplemented by a grid. In the right side of Figure 3 the Fisheye view still allows for an overview even while the focus has only a few items within it. This technique can be supplement by other visual means such as the use of a grid and/or shading as discussed by Zanella et al (Zanella et al., 2002).

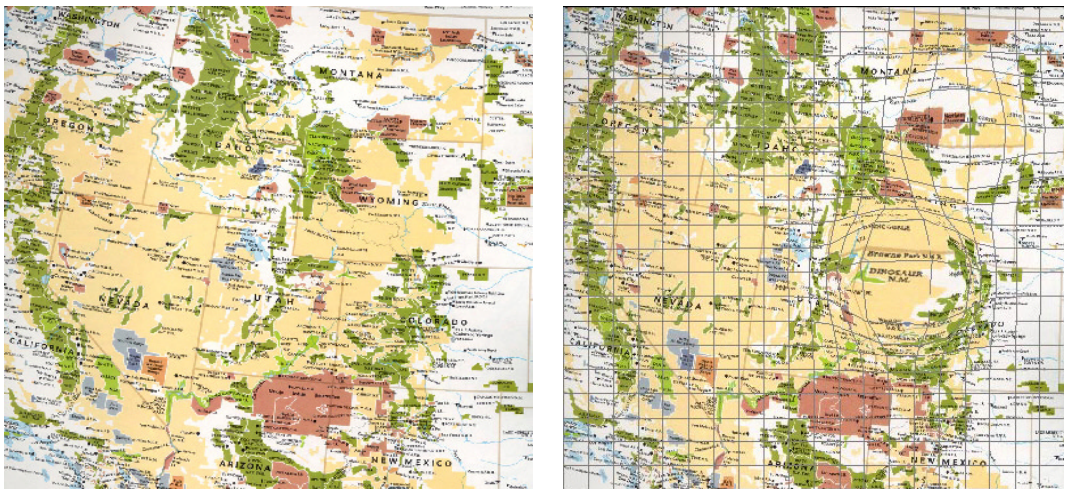


Figure 3: A Linear View versus A Fisheye View of the same map

(Source: A. Zanella, University of Calgary)

Advantages	Disadvantages
Highlights specific area's of interest while maintaining the overall context.	Does not improve the overall number of nodes which can be displayed

Table 1: Advantages and Disadvantages of Fisheye View

2.2.2 Hyperbolic Tree

Hyperbolic geometry is one of the non-Euclidean geometries developed at the turn of the century (Cannon et al., 1997). It is a focus and context view that lays out the hierarchy dataset in a uniform way on a hyperbolic plane and then maps this plane onto a circular display region. It allocates the same amount of room for each of the nodes in a tree while still avoiding collisions because there is an exponential amount of room available in hyperbolic space. While similar to the previous Fisheye technique and deriving elements from it, it is important to note the difference between the two is their use of geometries. The Fisheye view uses a normal Euclidean geometry whereas the Hyperbolic Tree uses a Hyperbolic geometry. This technique was pioneered at the Xerox Corporation by Lamping et al (1995). A spin-off company, Inxight now produces various software tools that employ this visualisation method. An example of the Inxight hyperbolic browser is shown in Figure 4. The hyperbolic tree browser developed at Xerox PARC was a 2D browser as opposed to later work that resulted in various 3D implementations such as the H3 browser, which is discussed in the next section. With a single still image, a projection from hyperbolic space looks similar to a Euclidean scene projected through a fisheye lens. The projection to hyperbolic space serves the purpose of a Degree of Interest function as required by Furnas (1986). The advantages of this technique are that any node of interest can be moved into the centre so its detail can be examined using simple dragging of the particular node. As this node becomes the focus of attention the entire tree is appropriately repositioned with this node as the root node. A node's context can be easily seen, as it may be viewed from all directions of the tree with its parent, siblings and children show in close proximity.

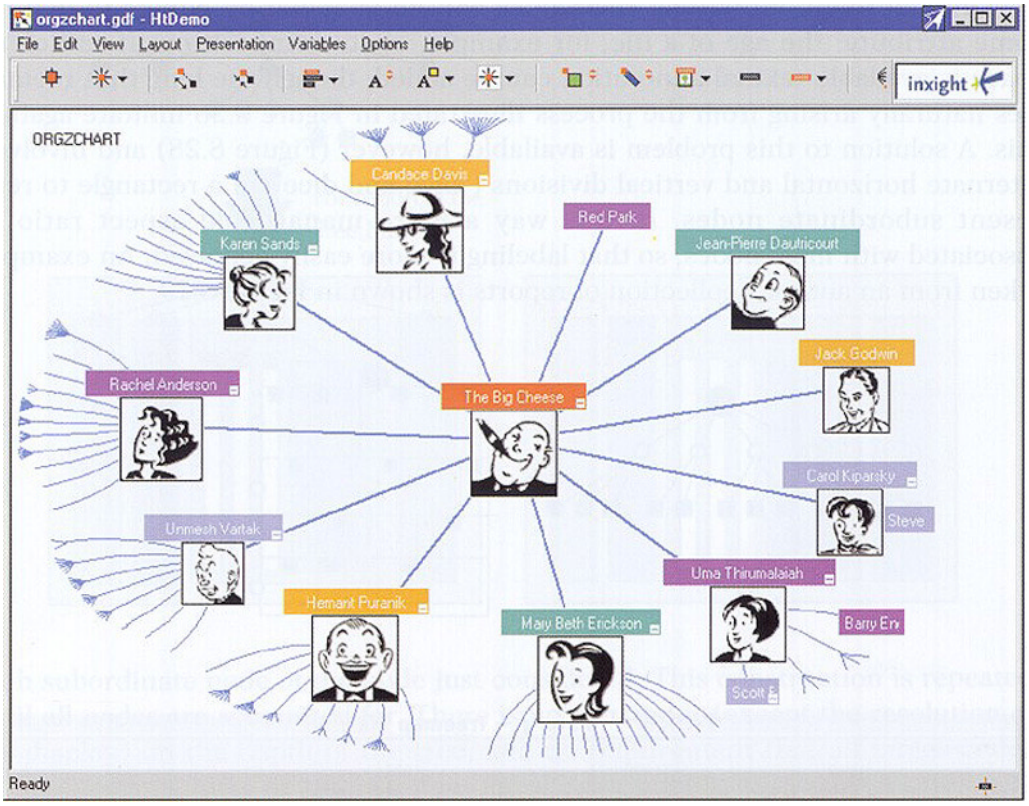


Figure 4: Inxight Hyperbolic Browser (©Inxight Corporation)

Advantages	Disadvantages
Similar to the Fisheye, it highlights specific area's of interest while maintaining the overall context.	Can only show dozens of labelled nodes (main nodes) on the screen Requires strictly hierarchical data

Table 2: Advantages and Disadvantages of Hyperbolic Tree

2.2.3 H3 Algorithm

The H3 layout algorithm (Munzner, 1997, Munzner, 2000) is derived from the hyperbolic browser and the cone tree (Robertson et al., 1991) systems from Xerox PARC. The Cone Tree algorithm lays out tree nodes hierarchically in 3D Euclidean space, with child nodes placed on the circumference of a cone emanating from the parent. Cone Trees are hierarchies laid out uniformly in three Nodes are drawn like 3x5 index cards. The top of the hierarchy is placed near the ceiling of the room, and is the apex of a cone with its children placed evenly spaced along its base. The next layer of nodes is drawn below the first, with their children in cones. The aspect ratio of the tree is fixed to fit the room. Each layer has cones of the same height (the room height divided by the tree depth). The H3 layout algorithm lays out child nodes on the surface of a hemisphere that covers the mouth of a cone, as Munzner describes it "like sprinkles on an ice cream cone" and is used for drawing large directed graphs as node-link diagrams in 3D hyperbolic space. The use of hyperbolic 3D space gives the advantage of the volume of the space increasing exponentially, rather than geometrically as occurs with 3D euclidean space.

The H3 also classifies the types of node within the graph as being either main, peripheral, and fringe. Main nodes are visible with shape and colour encodings; peripheral nodes are small but still distinguishable as individual nodes while fringe nodes are clustered together and can be seen on the extreme edges of the sphere. While this approach is similar to the hyperbolic browser there is a vast difference in the number of nodes that can be shown by both with the hyperbolic browser showing thousands of nodes and the H3 algorithm can easily deal with upwards of twenty thousand nodes, an example of this in practice can be seen in Figure 5. The XML3D, a novel browser that integrates an H3-based interactive 3D hyperbolic graph view with a more traditional 2D list view of the data was built to show the H3 algorithm was most useful when effectively linked with other interface components. Empirical testing by Risen et al (2000) found that this was indeed so.

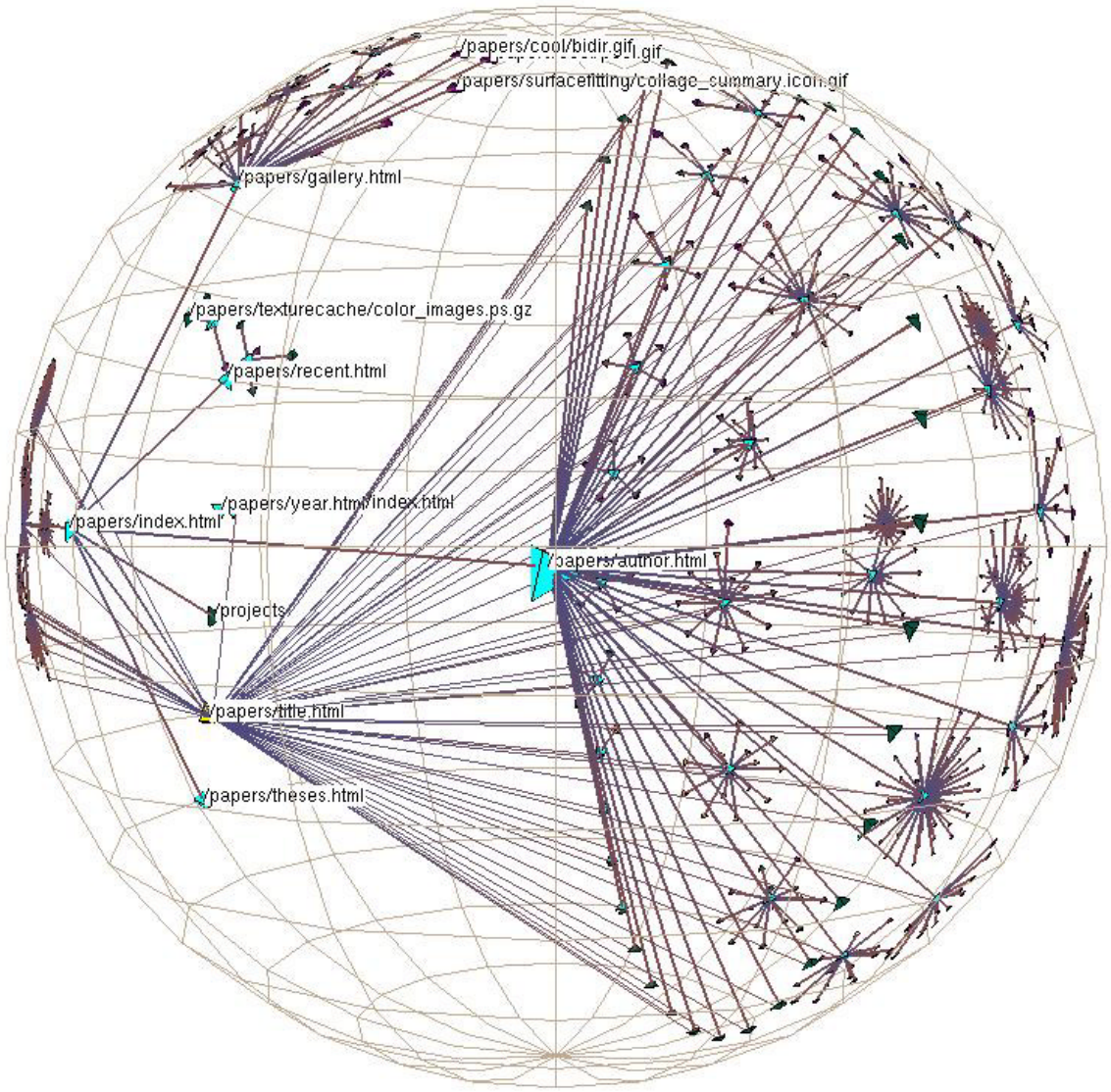


Figure 5: H3 Viewer representing a file system (Source: T. Munzner, Uni. Stanford)

Advantages	Disadvantages
Provides a three dimensional view of the dataset Can deal with thousands of entries Similar to the Fisheye and Hyperbolic Tree, it highlights specific area's of interest while maintaining the overall context.	Nodes at the periphery can be partially obscured

Table 3: Advantages and Disadvantages of H3 Algorithm

2.2.4 Tree maps

The Treemap is a space-filling layout that is generated automatically, used primarily for overviews of document collections and their meta-data. It was developed by Johnson and Shneiderman (1991, 1992b) and uses a rectangular, "slice and dice" methodology for representing hierarchical information structures as shown below in Figure 6. The "slice and dice" algorithm uses the entire screen to represent the tree root and its children. Each child of the root is given a horizontal or vertical strip of size proportional to the cumulative size of its descendents. The process is repeated recursively, flipping horizontal and vertical strips at each hierarchy level. Each of the strips or rectangular areas of alternate horizontal and vertical divisions are used to represent subordinate nodes. The rectangles are colour coded to some object attribute to achieve the rectangular method used to display information objects. These rectangles can have associated text or labelling describing the information object it represents. The Treemap creation can be easily explained from Figure 6 where the conversion from a simple tree structure to the related Treemap is shown. The Treemap has no constraint on the depth of a tree except of the resolution of the screen, and no requirement that all the nodes be at the same level or that the number of children of each node be the same.

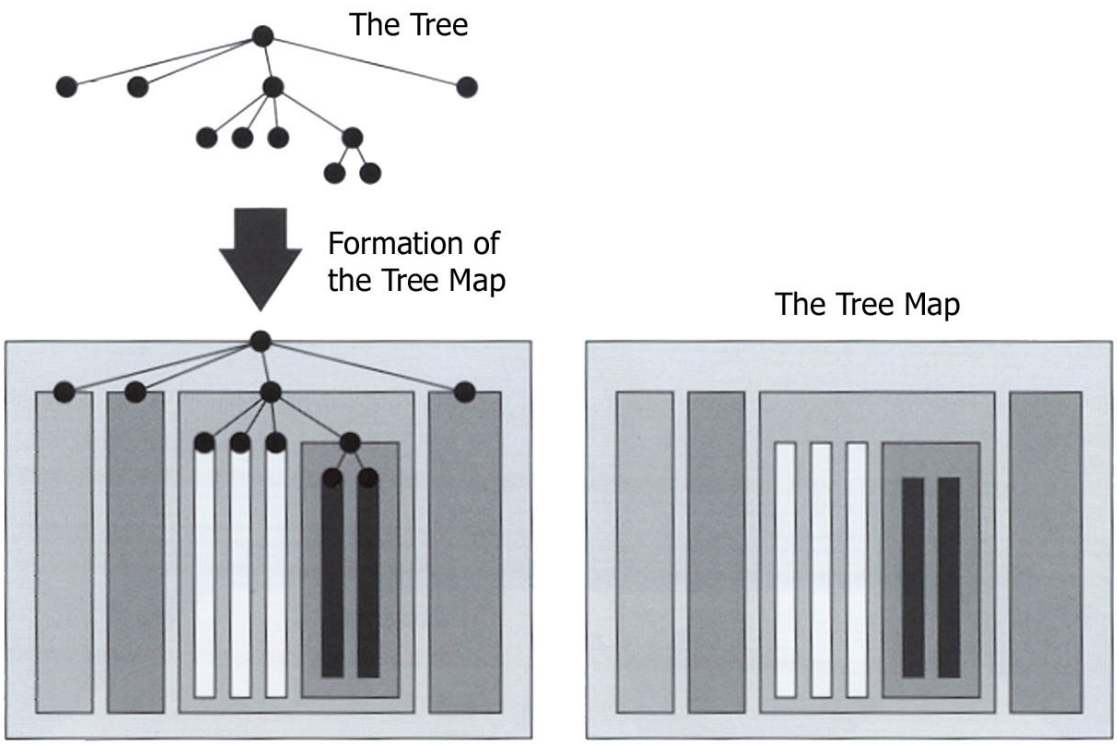


Figure 6: The Conversion of a tree to a Treemap (Source: Ben Shneiderman)

A disadvantage of the Treemap is that while being good in representing the attribute of the information structure portrayed through rectangle area (usually size), it is not so good at conveying the structure of the hierarchy. The Treemap method can be used to display any hierarchical data but one very typical application has been the display of an entire hard disk. In this application each file is represented as a rectangle with an area proportional to its file size and the colour used to represent the file type. The location can be mapped to either the type of information or the directory structure of the disk as shown in Figure 7.

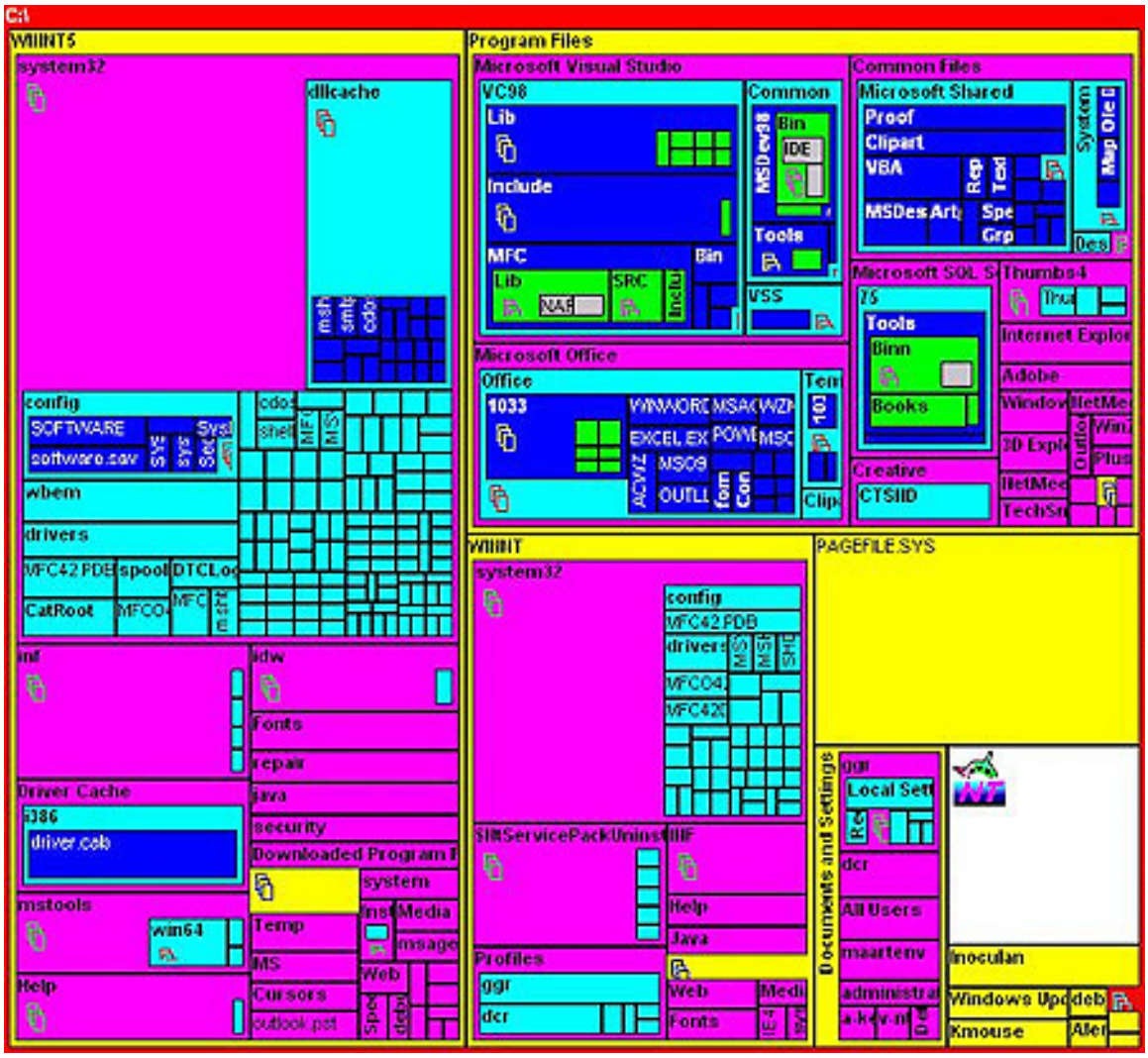


Figure 7: A TreeMap representing a file system (©MicroLogic DiskMapper)

Advantages	Disadvantages
Provides an overview of the dataset Offers a range of algorithms which use space effectively when displaying objects in particular circumstances	Requires well structure hierarchical data to provide a useful view of the dataset Can only effectively deal with a few thousand entries

Table 4: Advantages and Disadvantages of TreeMap

2.2.5 Starfield Displays

The Starfield display stems from prior scatterplot work by Williamson and Shneiderman (1992) on dynamic queries. Starfield display is essentially a two-dimensional scatter plot using structured result sets and zooming to reduce clutter. Dynamic queries allow for the querying a database and immediately return the desired information while maintaining a consistent representation of the information. Dynamic queries can be controlled by user interface components such as sliders. Dynamic queries allow users to rapidly explore and gain feedback from a display in a few tenths of a second. The major elements of difference between an ordinary query and a dynamic query are its direct manipulative nature, searching is rewarded with immediate feedback, and these elements are linked to a consistent display of the query results. The results are generally displayed using a graphical environment appropriate for the particular task domain (Williamson and Shneiderman, 1992). Ahlberg and Shneiderman (1993b) discuss dynamic queries in the context of visual information seeking (VIS) which is a methodology specially created to support visual browsing tasks and based on the work of Marchionini on information seeking tasks (1995). Visual information seeking is built upon the principles of direct manipulation (Shneiderman, 1992a):

- visual representation of the world of action including both the objects and actions
- rapid, incremental and reversible actions
- selection by pointing (not typing)
- immediate and continuous display of results

and then adds the following principles:

- dynamic query filters: query parameters are rapidly adjusted with sliders, buttons, etc.
- starfield display: result sets are continuously available and support viewing of hundreds or thousands of items
- tight coupling: query components are interrelated in ways that preserve display invariants and support progressive refinement. Specifically, outputs of queries can be easily used as input to produce other queries.

The Starfield display provides an overview of the dataset with the information objects being represented by coloured dots or shapes. The properties of the information objects are used to map to the screen location, colour and shape for each of the objects and its related graphical representation. The number of visible objects is controlled through the use of dynamic queries or by use of zoom mechanisms. The Filmfinder (Ahlberg and Shneiderman, 1994b) is an example of this type of display and is shown in Figure 8. Each dot in the left hand frame is an entry in a database about movies. The X-axis represents time and the Y-axis a measure of popularity with colour representing the movie's genre. To obtain more information about a particular element of the query results, users can double-click on that dot, getting the movie's details ("details-on-demand") and is shown in Figure 8 as the movie "The Witches of Eastwick". There are a number of Alphasliders (Ahlberg and Shneiderman, 1994a) used in this application and this interface component will be discussed in the next section. This visualisation work has been commercialised into the Spotfire product[5], which can be seen in Figure 9.

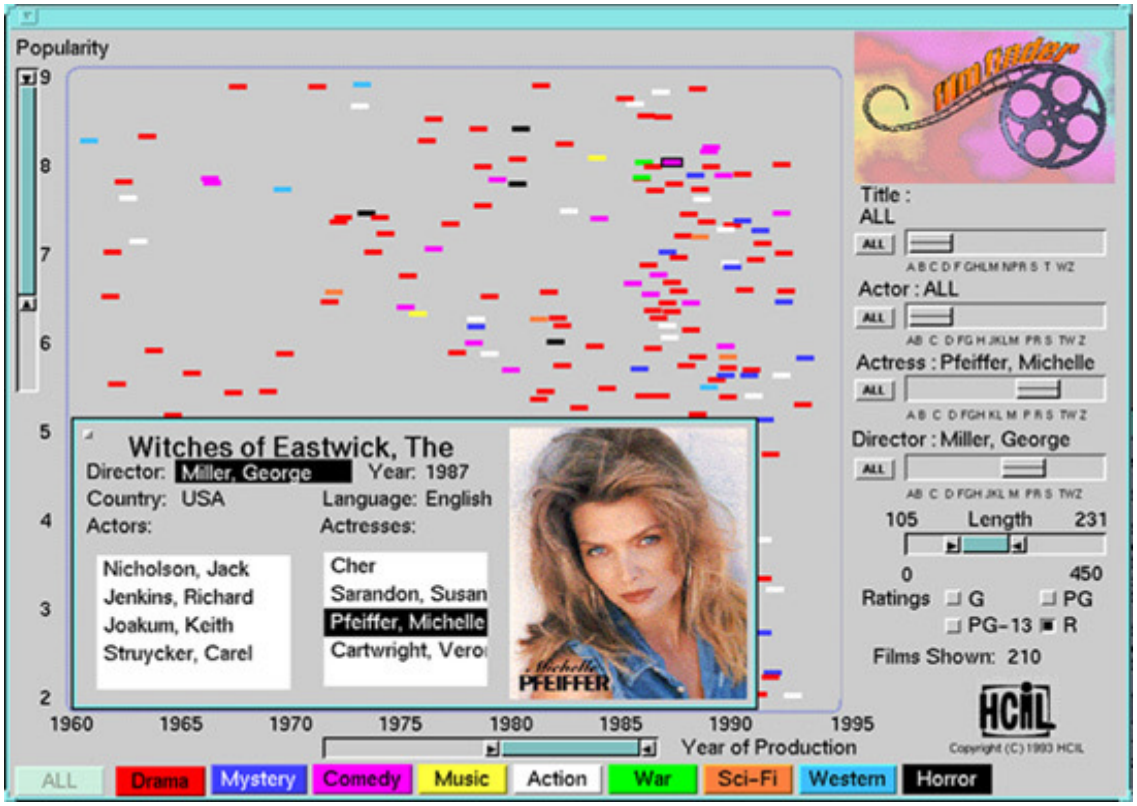


Figure 8: A Starfield Display, the Filmfinder (Source: HCIL, Uni. Maryland)

[5] http://www.spotfire.com

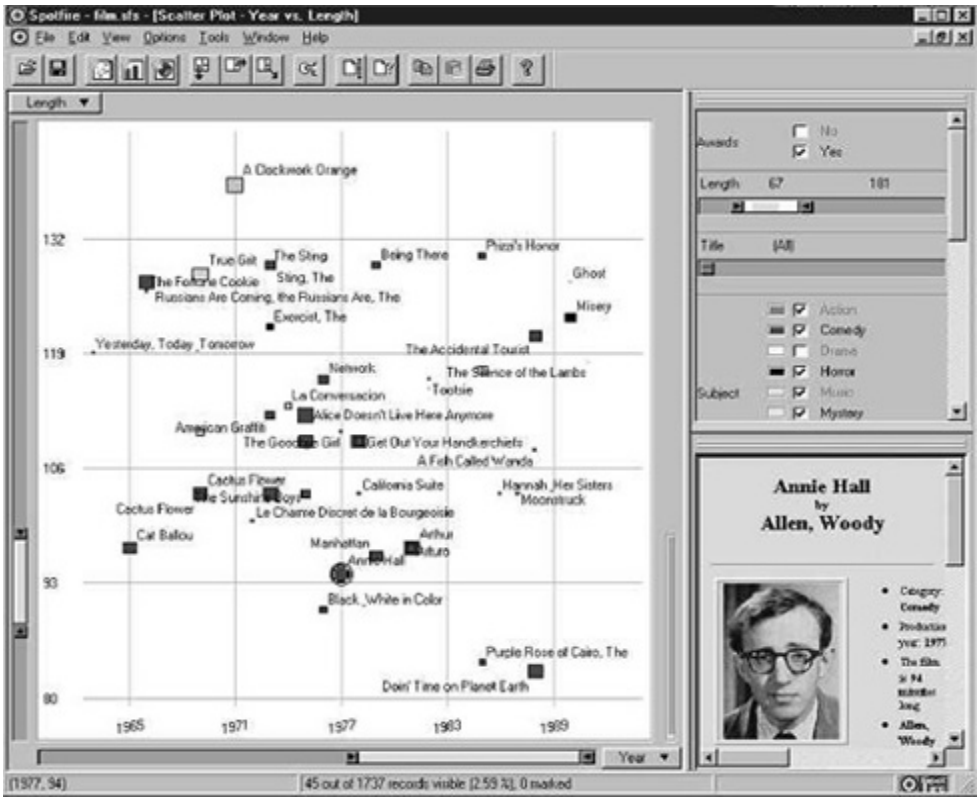


Figure 9: A Starfield Display, the Spotfire (© Spotfire AB)

Advantages	Disadvantages
Provides overview of a dataset With dynamic queries or zooming mechanisms can reduce visual clutter	Requires additional mechanisms such as dynamic queries or zooming to be effective

Table 5: Advantages and Disadvantages of Starfield Display

2.2.6 Zoomable User Interfaces

Zoomable User Interfaces (ZUIs) are based on alternative user interface paradigm using zooming and a single large information surface. The earlier foundations for ZUIs are found in work by Furnas and Bederson on multiscale interfaces called Space Scale Diagrams (1995). Space Scale Diagrams are at the theoretical core of ZUIs, this is where a diagram is created by many copies of an original 2D picture, one at each possible magnification, and stacking them up to form an inverted pyramid (Figure 10). While the horizontal axes represent the original spatial dimensions, the vertical axis represents scale, i.e., the magnification of the picture at that level. In theory, this representation is continuous and infinite: all magnifications appear from 0 to infinity, not just the four levels shown. ZUIs present information graphically and exploit people's innate spatial abilities. Detail can be shown without losing context, since the user can always rediscover context by zooming out. Zooming can be categorised as either *geometric* or *semantic*. Geometric zooming is where all the objects change only in their size so it simply provides a blown up vision of the dataset. Semantic zooming is where the objects change not only in size but also in detail so by zooming in on a node progressively more detail about that node is displayed.

The ZoomWorld application demonstrates the semantic zooming method, it is an interface for accessing a health board / district records. This application developed by Apricus was originally designed to computerise medical charts in an intensive care unit (ICU) of a hospital but when implemented using the ZUI paradigm was able to serve as an enterprise scale interface. This is illustrated by the ZoomWorld application (Raskin, 2000b) in Figure 11a –11d. The sequence shown in Figures 11a-11d shows the search process for a particular patients medial chart, starting with the desired hospital and zooming down with greater detail to the particular floor, ward, patient and finally the particular chart desired. Practical applications using the ZUI paradigm include Pad/Pad++ (Bederson and Hollan, 1994), JAZZ (Bederson and McAlister, 1999) and Zomit (Pook et al., 1998). Disorientation of users and the feeling of being "lost in hyperspace" have lead to the development of additions to ZUIs. Pook et al. (2000a) have created several orientation aids for ZUIs including a *context layer*, a *history layer*, a *hierarchy tree* and a

new pop-up menu called a *control menu* (2000b). This navigation approach aims to deal with both the large size of this type of database, while providing an engaging, interactive application allowing for discovery of new relationships within the data.

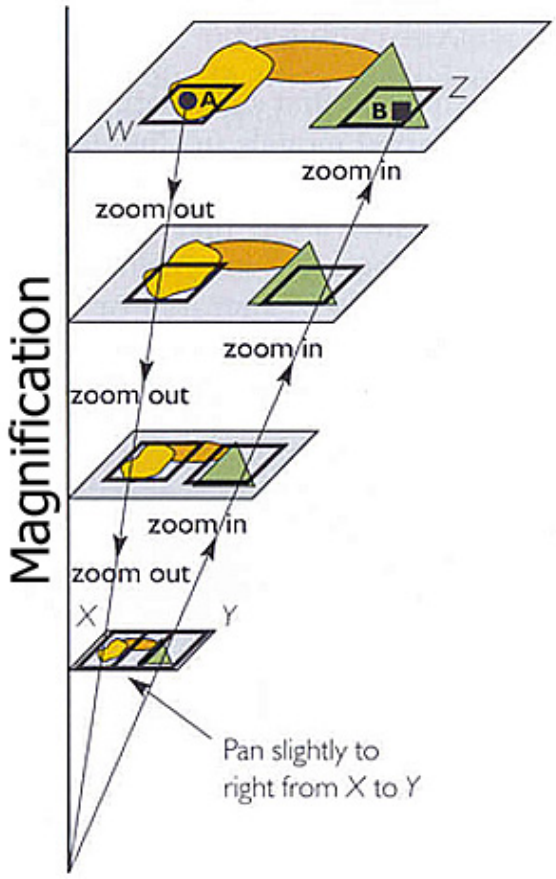


Figure 10: Space Scale Diagram (Source: Robert Spence)

Advantages	Disadvantages
Offers means to explore large datasets of information. Semantic zooming can be particularly useful and informative for user.	Context can be difficult to determine for use.

Table 6: Advantages and Disadvantages of Zoomable User Interfaces

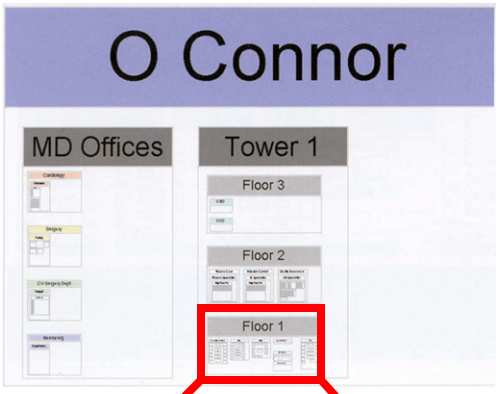


Figure 11a: Zoomworld Overview

(Source: Jef Raskin)

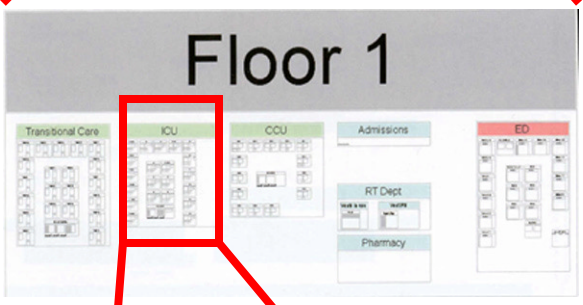


Figure 11b: Zoomworld Zoom Level 1

(Source: Jef Raskin)

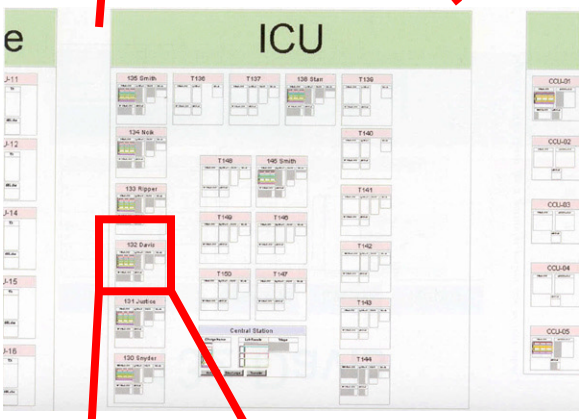


Figure 11c: Zoomworld Zoom Level 2

(Source: Jef Raskin)

Geometric zoom & Semantic zoom

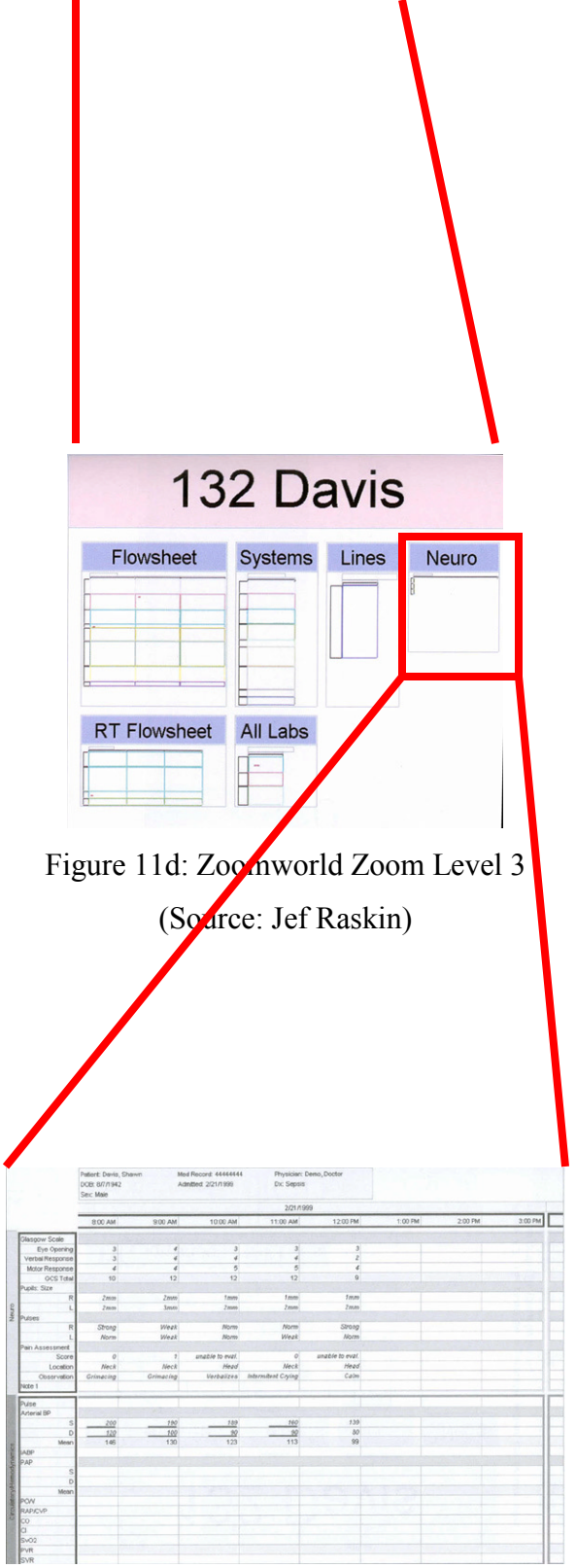


Figure 11d: Zoomworld Zoom Level 3

(Source: Jef Raskin)

Figure 11d: Zoomworld Zoom Details (Source: Jef Raskin)

2.2.7 Clustering

Cluster analysis is the process where entities are partitioned into groups based on the given features of each entity so that the groups are homogeneous and well separated. Each of these groups is called a *cluster*, and the method used is called the *clustering* algorithm. Clustering algorithms can be roughly divided into two categories: partitioning and hierarchical methods. Sprenger et al (2000) p2 define partitioning cluster methods (PCM) as "*attempt to analytically subdivide a set of data objects into a certain number of clusters*" and hierarchical clustering methods (HCM) as generating "a nested sequence of partitions".

Clustering can be applied in a multitude of domains ranging from economics, psychology to music and many others besides. Good introductions for clustering include Hartigan (1974), Gordan (1981), Murtagh (1985), McLachlan and Basford (1988) and Kaufman and Rouseeuw (1990). Hierarchical clustering algorithms produce a nested sequence of clusters, with a single all-inclusive cluster at the top and single point clusters at the bottom. The major focus of clustering in visualization regards it as a layout problem with the resulting solutions focusing on optimising the computation and spatial grouping of crowds of single data objects. This type of solution is based on partitioning clustering algorithms and can be seen in the *TimbreSpace Browser* (Tzanetakis and Cook, 2000c) application shown in Figure 12. The *H-BLOB (Hierarchical BLOB) algorithm* (Sprenger et al., 2000) is based on hierarchical clustering and can be seen in the document retrieval system shown in Figure 13. IVORY (Sprenger et al., 1998) is a Java toolkit for physics-based visualisation whose interface uses clustering as an approach for the display of multidimensional data.

The various clustering algorithms are a useful method for reducing multidimensional data and can be used to reduce the visual clutter on a display by reducing the number of elements present. Clustering can also be applied to gather data with similar properties such as musical songs from the same genre. Several audio techniques and applications, which apply this type of principle, are discussed in the next section on auditory browsing. As mentioned previously, this area is not the focus of our

research but it does contribute to an understanding of the available method for reducing visual clutter on a display by the application of various clustering algorithms.

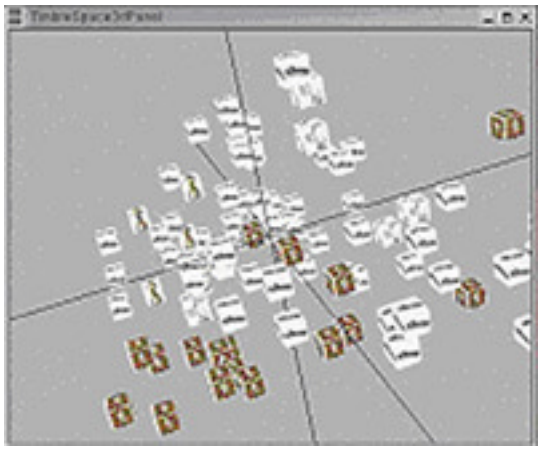


Figure 12: The *TimbreSpace Browser* (Source: George Tzanetakis)

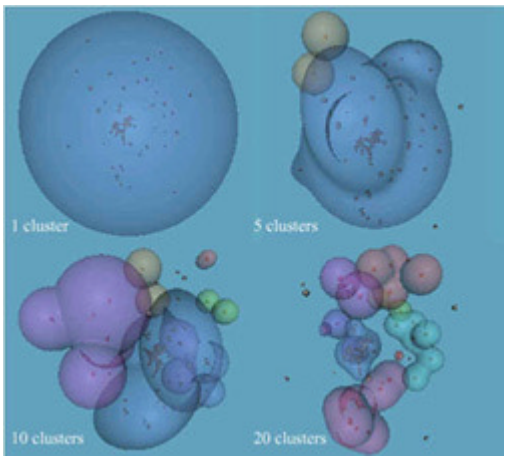


Figure 13: Example of H-BLOB clustering with cluster's of size 1,5,10 and 20 (Source: T. C. Sprenger)

Advantages	Disadvantages
Offers means to explore large datasets of information. Clustering of related objects can be useful and informative for user.	Clustering of objects require contextual knowledge of object

Table 7: Advantages and Disadvantages of Clustering

2.3 Interactive Widgets

Two particular user interface components or widgets are discussed due to their influences on our design choices for dynamic query controls. The two user interface components or widgets are the *Alphaslider* and the *Lensbar*. The *Alphaslider* (Ahlberg and Shneiderman, 1993a) is a derivative of the scrollbar widget that incorporates an index for the dataset and the ability to set the range of interest in the dataset dependant on some object property. The *Lensbar* (Masui, 1998) is another derivative of the scrollbar user interface widget with the addition of zooming and filtering.

The important difference between the visual browsing techniques and the user interface components (widgets) demonstrating them are the techniques provide a visualisation of a given dataset where the widgets are used as controls to supplement the control of the view. Awareness of the differences between visualisation and control allow for the creation of applications, which use these techniques and widgets to complement each other improving the usability and interactivity of an application.

2.3.1 Alphaslider

The first of the interactive widgets that we reviewed was a derivative of a computer slider. Computer sliders or scroll bars are a generic user input widget for specifying a numeric value from a range. Sliders are used to restrict the information portrayed on the screen or in essence filter it, thereby pruning the visual clutter and enabling only the desired range of objects to be displayed. One technique that has been used to apply dynamic queries to object collections is the Alphaslider (Ahlberg and Shneiderman, 1994a). A dynamic query is an approach allowing a user to search a dataset in a task domain using a direct manipulative approach. The dynamic query allows for the results to be immediately returned while still providing a consistent display of the dataset. A illustration of a dynamic query compared to a query can be seen using the Standard Query Language (SQL) for database queries. In our system, if we wished to find all the sounds with the music property of Jazz using only the database the resulting SQL query would be "Select * from Music where Description like 'Jazz';". As our system also offers dynamic queries, we only need to select the music slider and drag it to the Jazz entry to return all the sounds that are Jazz sounds. The difference between the database query and the dynamic query is the tight coupling between the interface and the dynamic query where the user is completely unaware of any databases and merely uses a slider to find the desired sounds. Another similar comparison can be made, with similar previous advantages for dynamic queries suggested when range of items are queried so that for example all the sounds from Classical to Salsa are returned.

Alphasliders are implemented as slider widgets but may refer to any mechanism used for navigating a large number of ordered objects using dynamic querying. In the particular instance of the slider widgets the scale range corresponds to the alphabetical range in which the items exist. Alphasliders mechanisms support selection from long lists of information using multigranular selection. Using the slider widget example again, an alphabetic index is provided next to the slider trough to show where specific ranges of a text space can be found, and the thumb provides different granularities for scrolling as shown in Figure 14. This range setting allows for the choice of all the entries between D

to S and scrolling within them as shown in Figure 14. The dynamic slider mechanisms used in the Sonic Browser are based this research.

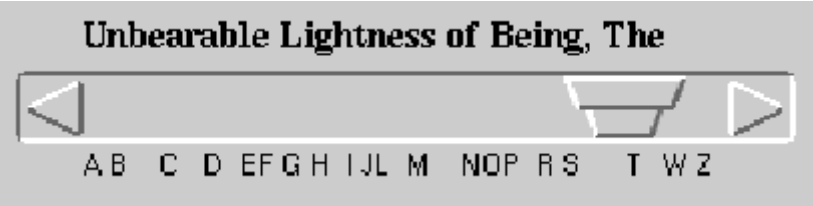


Figure 14: A close-up of an Alphaslider (Source: Ben Shneiderman)

Advantages	Disadvantages
Dynamic interface method Multigranular selection within a dataset	Requires structured data or classification of data for effectiveness

Table 8: Advantages and Disadvantages of Alphaslider

2.3.2 Lensbar

The Lensbar (Masui, 1998) is a further refinement of sliders (scroll bars) based on the Alphaslider but incorporating a filtering and zooming mechanism. This mechanism complements dynamic queries with approximate string matching, a zoom interface and a filtering mechanism. An example of the various Lensbar mechanisms and used as an interface to a dictionary is shown below in Figure 15. The Lensbar in combination, with the earlier work on the Alphaslider, influenced our choice and design of filtering mechanisms used in the Sonic Browser. Dynamic sliders, in the context of our research are a filtering mechanisms operating on arbitrary user classifications of the sounds within a collection. This filtering mechanism uses dynamic querying on a textual classification of the sounds within a collection.

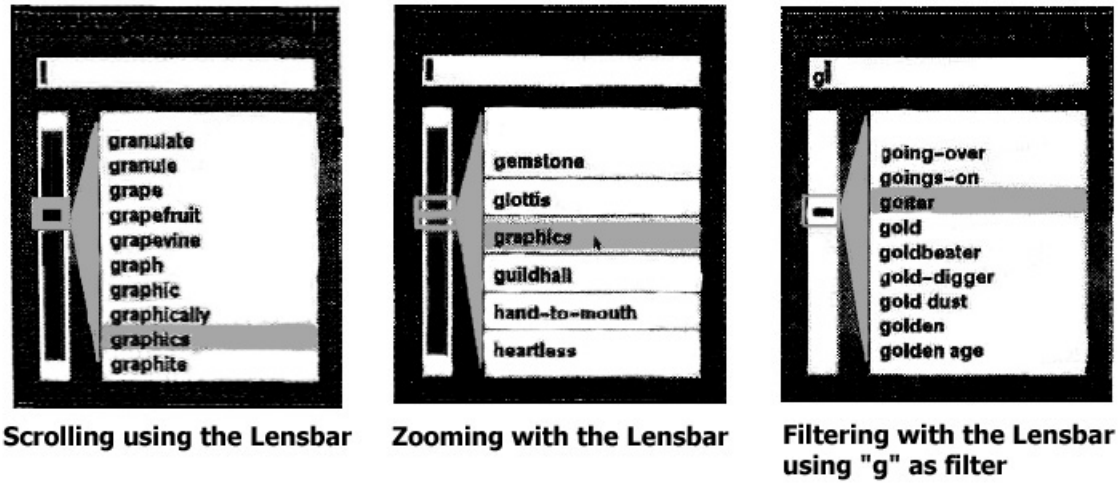


Figure 15: Using the Lensbar for I) scrolling, II) zooming and III) filtering (Source: T. Masui)

Advantages	Disadvantages
Combined dynamic manipulation interface with text retrieval methods	Requires structured data or classification of data for effectiveness

Table 9: Advantages and Disadvantages of Lensbar

2.4 Auditory Browsing

The different aspects of sound have provided several browsing systems for different applications. There are numerous different techniques and solutions available for the auditory browsing of data sets. The *thumbnailing* and *cue point* techniques mentioned in sections 2.4.3 and 2.4.4 are techniques of particular interest to this research. The common thread to these types of techniques is that they allow the user to focus on short segments or clips of a sound of interest that plays a significant part/s of the particular sound. In the following pages several of these techniques are reviewed.

A related application area is that of sound retrieval which has been spurred on by the now commonplace availability of digital audio resources such as MP3's to the home user. Using classification methods combined with visualisation techniques have shown improvements in the browsing and retrieval of audio resources. There are several examples of player, retrieval, browsing and classification applications discussed in Chapter 3, many of which use one or more of the techniques covered in this chapter.

2.4.1 Browsing with Sound Support

The Audible Web (Albers and Bergman, 1995) was the first web browser to add sound support for browsing text based web pages. The use of non-speech auditory cues (*auditory icons* see section 2.1) aided user monitoring of data transfer progress; provided feedback for user actions, and was used to provide auditory feedback while navigating of the WWW. The addition of sound to the interface was at a fairly low level of interactivity. The three tables below show the use of auditory cues in application Event's (Table 10), User Actions (Table 11) and Event/Attributes (Table 12). Albers also created hybrid systems using auditory icons and *sound spatialisation* (see section 2.3.4) to enhance operator performance in mission critical applications (Albers, 1994, Albers and Bergman, 1995, Albers, 1993). The *Varèse* system (Albers, 1995) was one such system which used a hybrid auditory interface in satellite ground control. In his research, Albers describes a methodology for creating hybrid auditory interfaces. These hybrid techniques can be applied to situations that cannot be handled by auditory icons, sonification or earcons alone. Research by Mynatt and Edwards (1992), added sound support for transparent access to X Windows applications with a particular focus on computer users who are blind or otherwise visually impaired. The interfaces created by Albers and Mynatt use sonification as interactive interface component / widget as opposed to sonifying the data directly. This highlights the two uses for sound support either as a method to provide an interface component or as a method to sonify the data being represented. The *loudSPIRE* system (LoPresti and Harris, 1996) added auditory display to a visualisation system which was used to support the navigation of data and event announcement, based on a hybrid auditory interface for navigation tasks. It uses three layers with earcons for system events, auditory icons for data elements and domain attributes were represented by themes of orchestral music and harmonious tonal sound that was parameterised by region and attribute value. These systems offer support for browsing via auditory displays but do not offer the browsing of audio resources or their content.

Event	Auditory Cue
Data Transfer	Pops & Clicks
Error Condition	Breaking Glass
Open External Program	Sliding/Opening

Table 10: Event / Auditory Cues

(Source: Michael Albers)

User Action	Auditory Cue
Press Button	Ca-Chick
Select Menu Item	Click
Scroll Up / Down	Short Pop
Select a Link	Bop-Bop

Table 11: User Actions / Auditory Cues

(Source: Michael Albers)

Event/Attribute	Auditory Cue
Relative Transfer Time	Tick-Tock
Error Condition	Breaking Glass
Pointer over Text File	Typewriter
Pointer over Graphics File	Camera Click
Pointer over Video File	16mm Movie Projector
Pointer over Audio File	Musical Piano Flourish
Pointer over Application File	Modem/Line Noise
File Size	Piano Notes

Table 12: Event/Attributes

(Source: Michael Albers)

Advantages	Disadvantages
Offers means to support notification and browsing tasks	Needs headphones or speakers Badly designed sound schemes of support can be irritating

Table 13: Advantages and Disadvantages of Browsing with Sound Support

2.4.2 Music Information Retrieval

The major focus of research in music information retrieval has focused mainly on the directed searching paradigm for collections as opposed to browsing of collections. There has been an increased interest in providing interface mechanisms for home use (Pauws et al., 2000) or car use (Amant et al., 2001). A typical example of research in music information retrieval is the *JRING* system for computer-assisted musicological analysis (Kornstädt, 2001), which uses information retrieval techniques and electronic formats to computerise the comparison task of the musicologist and is shown in Figure 16. The top most window in the screen shot displays the search parameters and the results are shown below in the next window. The search uses various parameters including a segment of a score. The results window shows the sounds containing this segment related to the search score with the specific segment highlighted in each of the result sounds. A browsing perspective on this area is found in (Fernström, 1998) and was the use scenario in the initial versions of the *Sonic Browser*. Other systems which to a degree support browsing for sound include systems such as *Muscle Fish* (Wold et al., 1996), *MARSYAS* (Tzanetakis and Cook, 2000c) and the *Ninja Web Jukebox* (Welsh et al., 1999). These systems, apart from the *Sonic Browser,* are mainly sound retrieval systems with some degree of browsing functionality but which is subordinate to the searching functionality. A detailed comparison of these browsing applications and also the MS Explorer is presented in section 3.1.

Advantages	Disadvantages
Focus on searching solutions using traditional information retrieval approaches	Specific focus on music sounds and the various forms of musical representation, which may not be suitable for other types of sounds.

Table 14: Advantages and Disadvantages of JRING, a typical MIR system

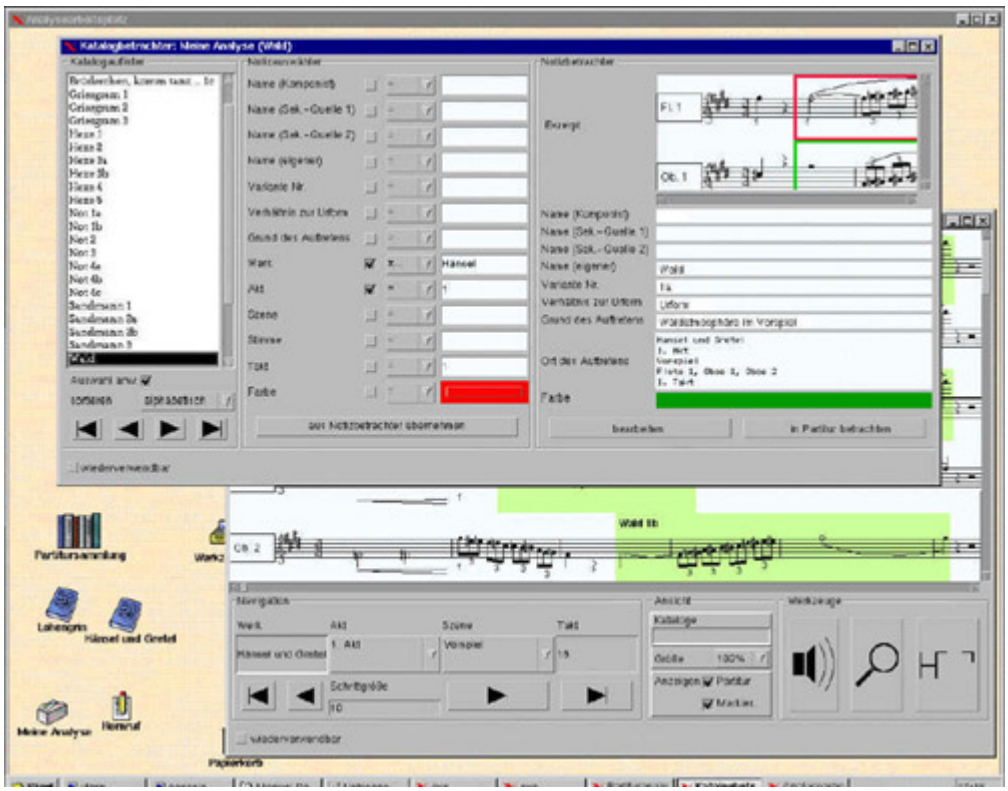


Figure 16: The *JRING* system for musicological analysis

(Source: Andreas Kornstädt)

2.4.3 Thumbnail

Thumbnail audio is a concept that has been adapted from image thumbnailing. Thumbnailing is defined as "the process of creating a short summary sound file from a large sound file in such a way that the summary best captures the essential elements of the original sound file" (Tzanetakis, 2002) p71. There are a couple of variations in its adaptation. The idea of *Timbregrams* (Tzanetakis and Cook, 2000a) use thumbnailing in the creation of a small graphic representing the timbral and temporal information of the sound file and exploits the human visual ability for colour perception and pattern recognition. Speech and classical music timbregrams are shown in Figure 17. The thumbnailing or audio scrubbing concept is where the audio file is thumbnailed into a small clip that contains a condensed version of the entire sound.

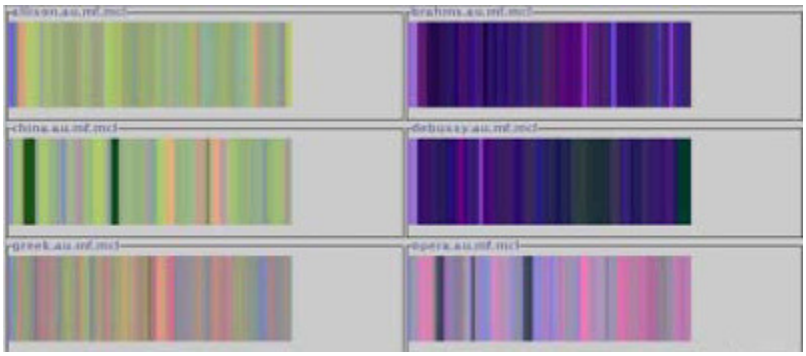


Figure 17: Timbregrams for speech and classical music

(Source: George Tzanetakis)

Advantages	Disadvantages
Summarises the essence of a sound using a short visual and/or auditory technique	Require pre-processing of a sound

Table 15: Advantages and Disadvantages of Thumbnail techniques

2.4.4 Cue point

Cue Points or marker points are pointers to important events in a sound file. Cue Points are the "meta data" in a file format (Prothman, 2000), which refer to points or sections of interest within the sound. For examples music files can have the chorus and/or an instrumental solo event marked as cue point. Research by Warren (1999) has shown that for recognition of sounds, is most often sufficient to hear only 500 ms to 2 seconds of the characteristic or significant part of a sound file. Exploiting this fact can lead to faster browsing of digital sound resources by only playing a short relevant section of a file. Cue Points are implemented in a variety of manners in the different file formats and an introduction to this aspect of file formats was previously investigated (Brazil, 2001). Another advantage of this mechanism is that silence which sometimes occurs at the start of a sound files may be avoided by using a cue point that is set to where the sound begins not where the file begins. Cue Points in sound browsing applications enable users to explore large numbers of sounds faster by allowing users to concentrate on the significant portion of the sound, which should allow for a faster recognition. Currently, we know of no sound browsing tool that has implemented this functionality except our prototype system. This excludes sound editing tools but these type of applications are not specially geared towards sound browsing.

Advantages	Disadvantages
Highlights a point/s or an area/s of interest within a sound file	Numerous different formats with different methods to achieve same results

Table 16: Advantages and Disadvantages of Cue point techniques

2.4.5 Spatial Sound

Spatialising a sound using the technique of sound localisation that employs artificial manipulation upon a single input channel to creates a spatialised sound. The sound can be presented over headphones or via loudspeakers and offers a good simulation of the sound as it would occur naturally. Head-Related Transfer Functions (HRTFs) are one method for creating the necessary auditory localization cues for spatialising a sound (Wenzel, 1994) and allow for a sound lateralisation around the head of the listener. Another method is stereo amplitude panning which allow for sound lateralisation to the left or right of the listener's head. The use of sound spatialisation has been used in various applications and domains, from cockpits to web browsing and audio diaries. Spatial sound was used by Begault (1994) to enhance speech perception in cockpits and mission control systems and showed that spatial position is one of the cues for auditory stream segregation (see Section 2.1). The *AudioStreamer* (Kobayashi and Schmandt, 1997) showed that the multiple stream speech perception is further enhanced by the use of sound spatialisation and the "cocktail party" effect as discussed in section 2.1. Walker et al (2001) explored the use of spatialised sound for use as an auditory interface for a diary program to overcome the problem of small screen space on portable digital assistant's (PDA's). A horizontal ('clock-face') display orientation on the horizontal plane was used as the auditory display space.

Advantages	Disadvantages
Generates a more realistic three dimensional environment	Requires more complex processing of a sound

Table 17: Advantages and Disadvantages of Spatial Sound

2.5 Review Of Suitable Techniques For Browsing Audio Collections

After reviewing the visualisation techniques in the context of finding suitable techniques for use as complementary visualisations of an audio collection, we selected four visualisation techniques. The techniques that were chosen were the TreeMap, the Hyperbolic Tree, a simple Zoomable User Interface and a Starfield display. The TreeMap was chosen to see how the "slice and dice" method (see Section 2.2.4) could be applied to sound resources, which have some hierarchical properties. The Hyperbolic Tree was chosen as it was felt it would provide an engaging view of the dataset. The ZUI was chosen for both for its availability via an open source software component and to investigate simple geometric zooming as a browsing mechanism. Our Starfield display derivative was based on the visualisation used in previous versions of the Sonic Browser. The Fisheye technique was not selected as it was felt to be already encompassed by the Hyperbolic Tree visualisation and the H3 algorithm was not selected as we felt that the Hyperbolic Tree would indicate if this type of technique was suitable. In seeking to implement a dynamic query control we drew upon influences of the interactive user interface widgets of the Alphaslider and the Lensbar. Simple clustering techniques were available for object layout but as the implementation of a classification / comparison framework was beyond the scope of this research more advanced clustering was not explored.

The review of the auditory techniques resulted in a browsing system with support for cue points but influenced by *MARSYAS, JRING* and other musical information retrieval systems. The choice of the other auditory techniques was based on previous techniques implemented in prior versions of the *Sonic Browser* such as multistream simultaneous audio playback with the addition of new techniques such as cue points.

2.6 Chapter Summary

This chapter has presented a review of the previous research in the areas of sonification and browsing. The chapter has discussed several systems, methods and techniques for the sonification and browsing of resources both graphical and auditory. The finding of this review of the available previous research creates the foundations for our research. The advantages of reviewing previous techniques ensures the use of the most appropriate techniques as well generating an awareness of the failings of previous systems and techniques.

The next chapter introduces the comparable systems that influenced the design of the Sonic Browser and elaborates upon the particular advantages and disadvantages of the comparable systems. The previous work on the Sonic Browser is also discussed. A summary of the insights gained from examining all these systems concluded the chapter.

Chapter 3

Audio Browsing Systems

3 Audio Browsing Systems

In the area of Sonic Browsing there are only a handful of systems that are comparable to this research. Here follows a brief overview of each, outlining the interface and the system's advantages & disadvantages. At the end of this section, we present a summary of the systems and the lessons learnt from their implementations.

3.1 Microsoft Explorer

Perhaps the most common interface used for the browsing of sounds is that of the Windows File explorer. This often used as the standard GUI for navigating a file system within a Microsoft Windows operating system and it should be noted that it was not intended as an audio browser tool but as a general file management tool. It is included in this review due its common usage as an audio browsing tool. The historical basis for this type of file system navigator derives from a standard approach taken by the Macintosh operating systems when dealing with the Apple Macintosh Hierarchical File System, which presents the user with the file system as a folder/file hierarchy. It offers several advantages such as multiple views of the data such as Icon, List, Detail and Thumbnail and automatically recognises common media and associates the related application or player to the file. This allows a user to simply double-click a sound file to get it to play in the relevant player. There are three disadvantages to this interface with regard to browsing sound. Firstly, one chooses to play the file explicitly using a double-click selection, secondly that the current mechanisms only allows for a single audio file to be played at a time. The third disadvantage are the mapping categories that are used in the interface are statically mapped as well as being mapped to properties of the sound file within the file system, not to its actual sound properties. A screenshot of the standard Windows MS Explorer is shown below in Figure 18. The view shown in Figure 18 corresponds to the Detail view available in this interface.

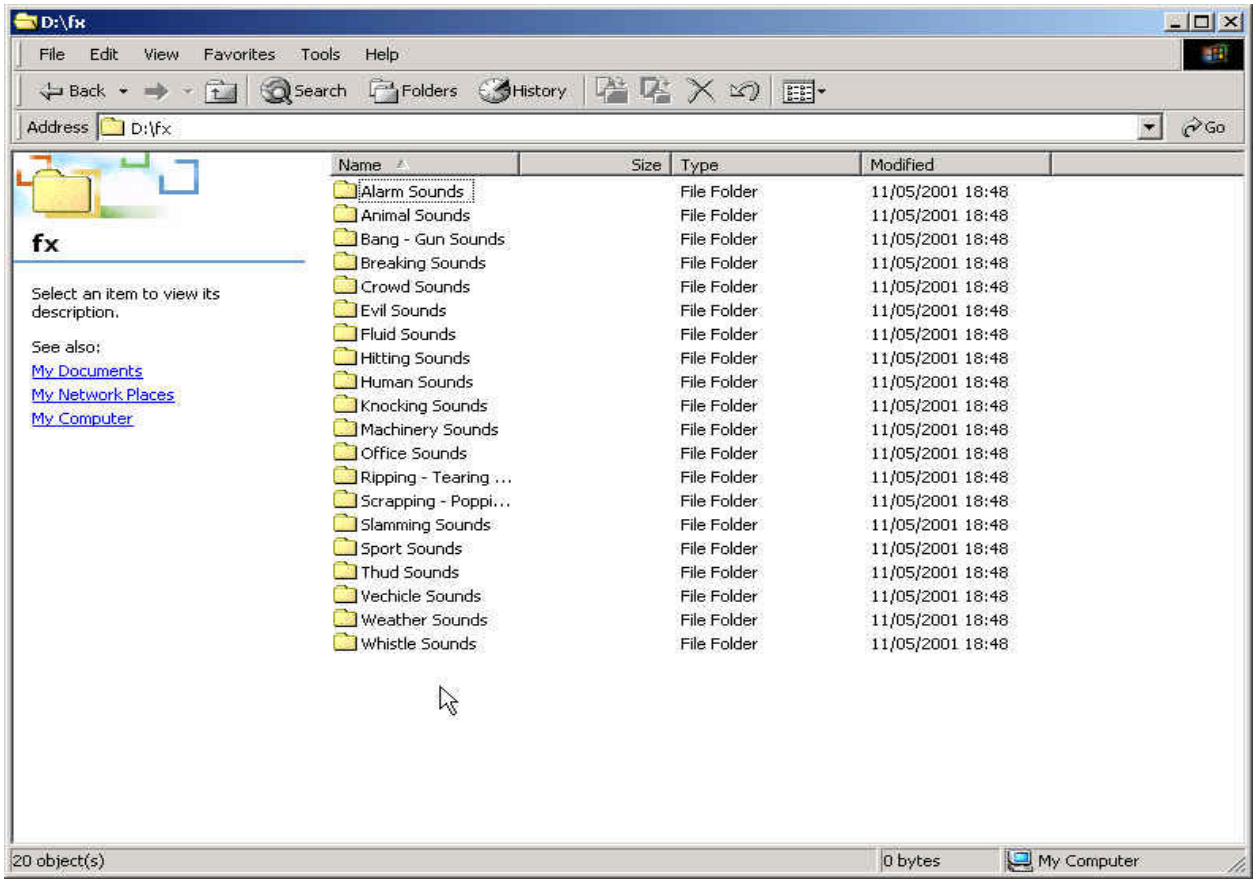


Figure 18: Windows MS Explorer

Advantages	Disadvantages
Familiar interface style	Single audio stream Player style interface Sound files are treated as files, no information regarding sound content is available

Table 18: Advantages and Disadvantages of the Windows MS Explorer

3.2 Nullsoft Winamp3®

Winamp3® is a freeware media player shown in Figure 19, which has been developed by Nullsoft Inc. It is a highly customisable sound and video player that offers many features over the standard audio players distributed with many operating systems. It offers an software development kit (SDK) for developers who wish to further modify the application. As a common player for users with MP3 collections, its query interface (Media Library) provides an example of what functionality is available in one of the more developed media players as shown in Figure 20. This system offers only a previous / next song browsing ability and is limited to song name, album or artist for properties that may be searched. It is a system specifically designed for playing music files and its functionality does not extend very well to other sound types such as speech or environmental audio. As it is based on a traditional player style interface with stop, play, pause, fast forward and rewind functionality it can only play a single audio file at a time.

Figure 19: The Winamp3 Player, main window (Source: Nullsoft, Inc.)

Figure 20: The Winamp3 Media Library (Source: Nullsoft, Inc.)

Advantages	Disadvantages
Well suited to playing music files	Single stream audio
Can play remote files via network / Internet	Player style interface – Play, Stop, Pause, Fast
Read music information from ID3 tags	Forward and Rewind

Table 19: Advantages and Disadvantages of the Winamp3 Player

3.3 Ninja Web Jukebox

The Ninja Jukebox uses networking to access sound collections but only offers a traditional music player interface as shown in the bottom of Figure 21. It does however offer much more advanced sound comparison by using several sound classification algorithms (Zhang, 1998a, Zhang, 1998b) to calculate sound similarity data, which is used in the search process. These algorithms are designed for musical sounds and function most effectively for the musical genres of classical and soul music. This is another tool, which is specifically designed for music and the similarity algorithms do not extend to other sound categories and it also only plays a single sound file at a time. The initial processing of the songs in this system can be quite time-consuming, however queries offer real or near real time performance until the dataset contains more than ten thousand songs after which performance degrades. The player interface is shown in the bottom Figure 21.

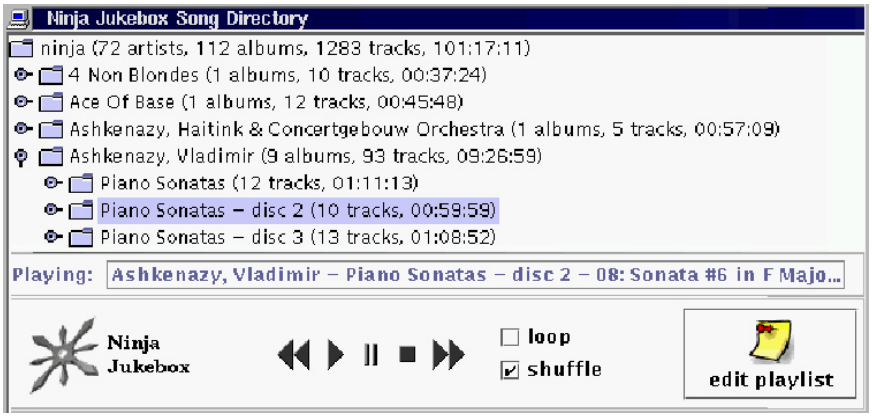


Figure 21: The Ninja Jukebox Client GUI

(Source: M. Welsh, Uni. Of California, Berkeley)

The search mechanism, which displays a list of similar songs based on the algorithmic comparison of sounds, is shown in Figure 22. The properties used in these algorithms are tonal histograms, tonal transitions, noise, volume, and tempo. These are discussed in greater detail by Welsh et al (1999).

Figure 22: The Ninja Jukebox Search GUI

(Source: M. Welsh, Uni. Of California, Berkeley)

Advantages	Disadvantages
Client – Server architecture	Single stream audio
Sound comparison engine	Player style interface – Play, Stop, Pause, Fast Forward and Rewind

Table 20: Advantages and Disadvantages of the Ninja Jukebox

3.4 SoundFisher

SoundFisher is a front-end application to access a content-based classification system developed by Wold et al (1996). This interface was the first to offer retrieval of audio collection using properties of the sounds within collection such as simile, onomatopoeia, acoustic/perceptual features and subject features. These properties were combined with a keyword and text retrieval interface as shown in Figure 23. The visualisation aspects of this interface are poor and it is again limited to playing a single audio file. The latest commercial incarnation of this interface is called MuscleFish and is available online[6]. MuscleFish offers a redesigned GUI but it is still very similar to the traditional player controls and list arrangement.

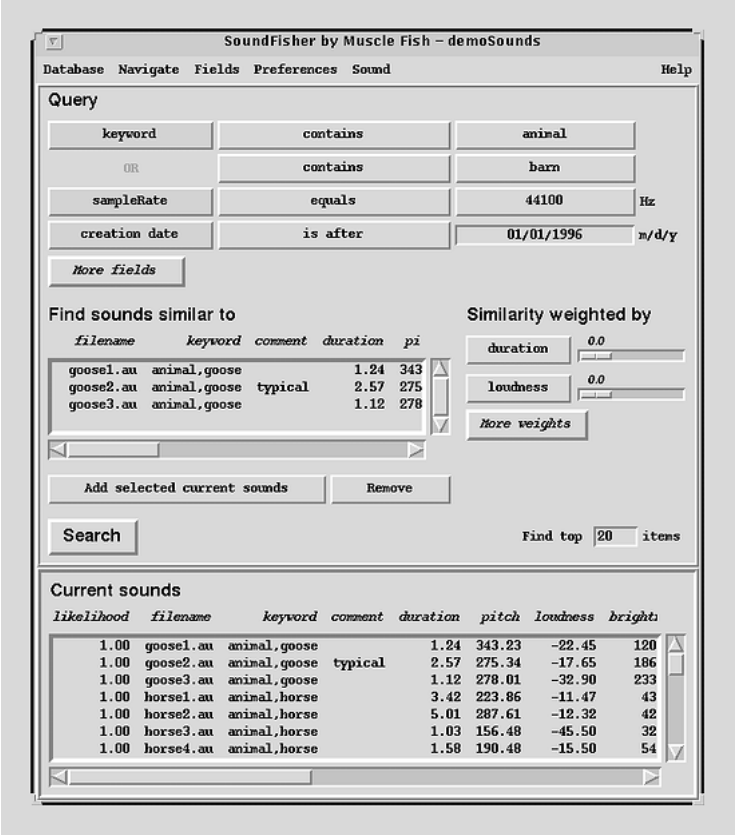


Figure 23: The SoundFisher GUI

(Source: E. Wold, Muscle Fish Inc.)

[6] http://www.musclefish.com

Advantages	Disadvantages
Similarity comparison engine New categories offered such as simile, onomatopoeia with both acoustic and subjective audio features	Information Retrieval style interface Single stream audio

Table 21: Advantages and Disadvantages of the SoundFisher

3.5 Marsyas

Marsyas (Tzanetakis and Cook, 2000c) is a software framework for analysis and classification of audio with a focus on the tools and mechanisms for these processes but it also offers two graphical interfaces. The first is the TimbreSpace browser, which has already been discussed in *2.2.7* as a visualisation technique for clustered datasets. The second graphical interface is the "enhanced audio browser" as shown in Figure 24. This work has been directed towards the analysis and classification of musical sounds, in particular, the musical genre of jazz. Tzanetakis reports the results of using this interface and of its classification methods (2002). Later versions of Marsyas have been combined with a Timbregrams visualisation of the sound files, as already discussed in section 2.4.3. The Marsyas framework is focused on analytical aspects of sounds but offers traditional player controls supplemented with zooming and tagging features.

The Audio Retrieval Browser (ARB) prototype system (Brazil et al., 2002a) has been investigated as the first exploratory application in this research, combining the previous work by the MARSYAS software framework and on the Sonic Browser. The goal was to provide an interactive application with automatic sound classification using the best features of the two previous applications, MARSYAS and the Sonic Browser. The Audio Retrieval Browser was the first prototype of this application. The ARB uses a flexible distributed graphical user interface with multiple visualization components combined with an automatic classification of musical sounds that provides novel ways of browsing large audio collections. The investigation examined a new interface, which would combine the best features of these two software systems. Marsyas offers many sound classification algorithms with single audio stream playback interface whereas the Sonic Browser offered a direct manipulation interface with multiple audio stream playback interface, combining these would create a hybrid with the best features of both. In both systems the central idea is to map sounds to aural and/or visual objects with properties that convey information about the audio resources and use these objects in order to create browsing spaces. The merging of these systems investigated a new graphical user interface, which would allow for the flexible manual and automatic setting of application parameters whilst providing a 2D or 2.5D soundscape with an interactive

aura playback of multiple stream audio. The results derived from this investigation have been incorporated into this research for our prototype system as discussed previously.

The next section deals with the previous work on the Sonic Browser and discusses the application and its features. The combination of the previously discussed applications and of the history of the development of the Sonic Browser, allows for an understanding of the state of the art prior to the completion of this research.

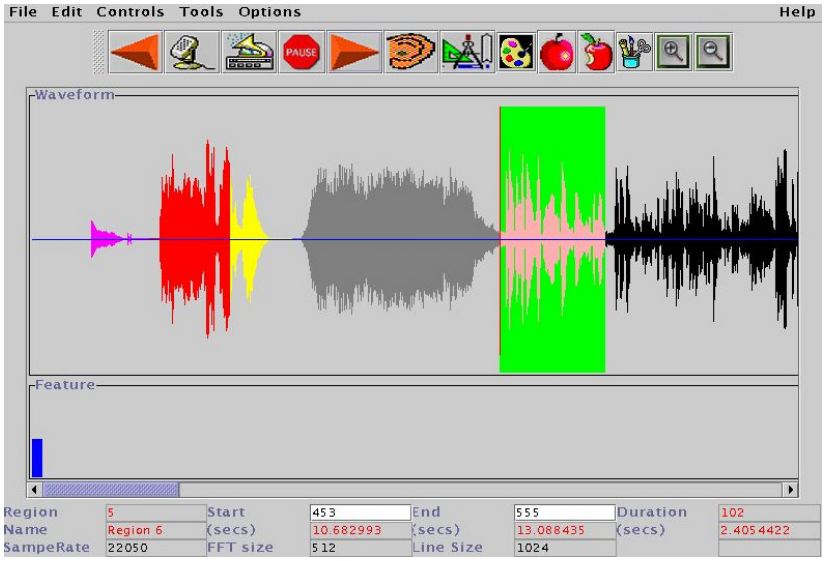


Figure 24: The Enhanced Audio Browser, a graphical interface to the Marsyas framework (Source: George Tzanetakis)

Advantages	Disadvantages
Detailed information on sound file	Single stream audio
Wave display	Player style interface – Play, Stop, Pause, Fast
Linked to analysis framework	Forward and Rewind

Table 22: Advantages and Disadvantages of the Marsyas framework

3.6 Sonic Browser

The foundation of the research into the Sonic Browser was a previous investigation into "how to improve the interactive experience for people exploring large complex heterogeneous data sets" (Fernström and Bannon, 1997a). One comment from this previous investigation occurred when showing a musicologist the sonic browser prototype for the first time. He expressed great surprise exclaiming, "I've never seen a collection this way before". This is a very important comment as he had been working for a couple of years re-cataloguing a music collection, from index cards to database, via desktop publishing tools to its final form - a paper based product, ready to print. Still, in paper or data base format, one cannot get a visual spatial overview of, for example, 'here are the jigs and there are the reels'. This can be the case when musicologists are searching for melodies in a collection. Melodies collected through fieldwork can often be different to older original versions. They can still be the same melodies but with the addition of an individual performer's style. This sometimes makes it difficult to use computer-based search algorithms or formal queries. The scenario from Fernström's earlier investigations in MIR (1998), corresponds well to user needs as described by Alain Bonardi (2000), where he lists some key features for a musicologist's workstation. The particular scenario was concerned with when a musicologist had recorded a particular music piece from observation in the real world and was looking at the existing collection to see if the piece already existed in some form within the collection. If the particular piece did not exist it would be transcribed, collated and added to the collection but for Fernström's scenario the focus was on the browsing of the collection to see if the particular tune was present.

The first prototype or BROWSE system used the "star-field" display for representation of the data set (see Section 2.2.5 for more details). The reasoning behind this choice was that it would potentially offer the best possibilities for displaying the properties of tunes in the collection. Most of the properties in this particular data set are abstract entities and there is no definite hierarchical relation between tunes and between their properties. The particular mappings were discussed with several musicologists until the mapping in table 23 was deemed most suitable by the musicologists involved. The visual properties of objects studied were shape, size, colour and location with the relevant mapping as shown in table 23. The BROWSE system was based on a traditional WIMP

approach, as much as possible of the active screen area is made available to display a "star-field" visual representation of the data set. In addition to the "star-field", a number of functions were required: menus for general control of the browser, a tool bar for widgets and controls, and a status area to provide working memory support for the user for the visual representations in the main viewing area as shown in Figure 25. When the user moved the cursor in the starfield, the tune objects (represented by the shapes) closest to the cursor would start to play. When the cursor remained static on a particular shape, only the related tune object would play, centred in auditory spatial location. The mappings can be seen in Figure 25 with the squares representing jigs, the circles representing reels; red indicates a violin, blue indicates an accordion and so on, with the X, Y and size graphical properties being represented by the chronology, mode and length of tune respectively.

Property	Representation
Chronology	Horizontal location
Mode	Vertical location
Main Category	Colour
Sub Category	Shape
Length of tune	Size of shape

Table 23: Mapping of properties of the data set to visual representations

The BROWSE system concentrated on investigating the combination of the visual and auditory modalities. The tight coupling between the visual and sonic representation of objects allowed users to move the cursor around the application to get an impression of the *sonic neighbourhood*. It was also found that musical object's properties could support differentiation. Another facet was the ability of the user to configure all aspect of the sonic representation. The BROWSE system is shown in Figure 29. The BROWSE system was later incorporated into a new system, the *Sonic Browser*.

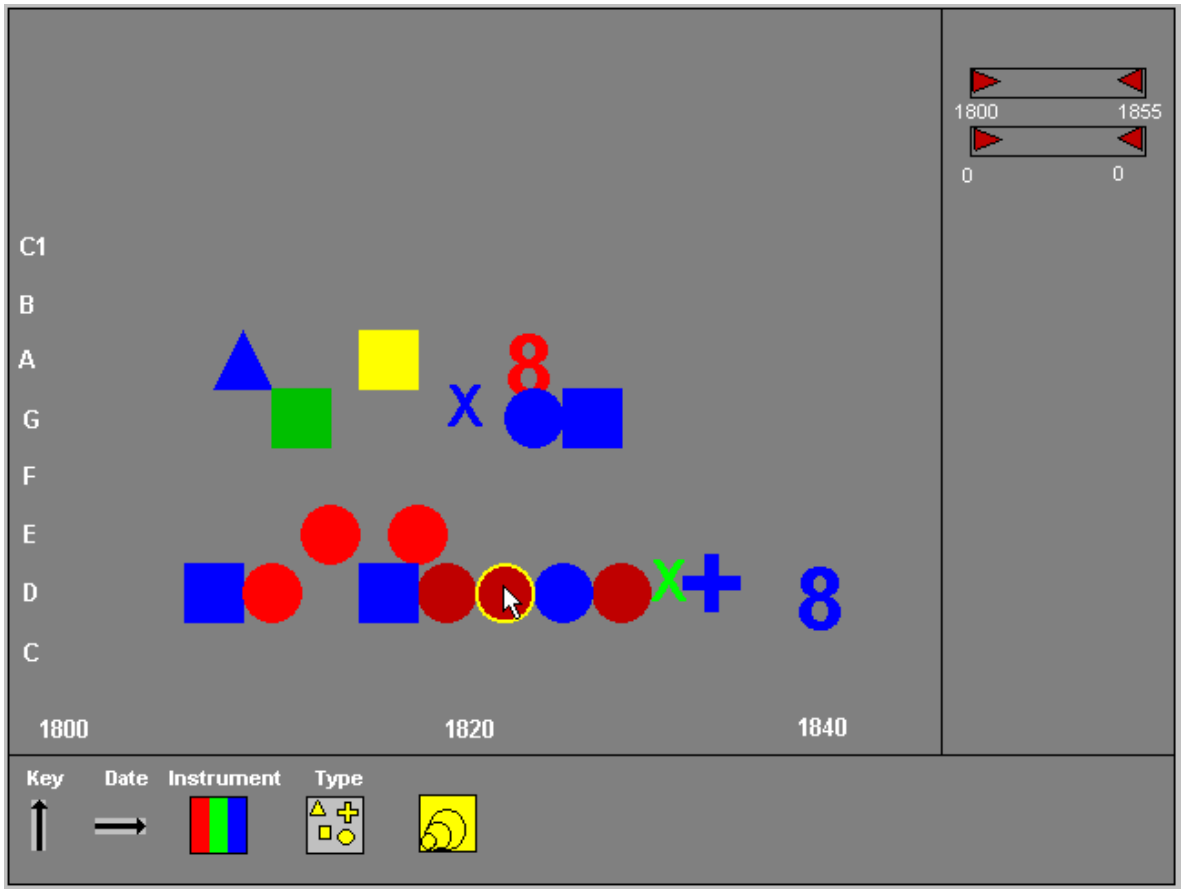


Figure 25: BROWSE System

Advantages	Disadvantages
Used a “star-field” representation Multiple properties of object being mapped to visual object Sound played by cursor over	Single stream audio playback

Table 24: Advantages and Disadvantages of the BROWSE system

The first version of the Sonic Browser by Fernström was designed and implemented in C++ and is shown in Figure 26. The property mappings and function areas correspond to those available in the previous BROWSE system. The symbols used for representing properties were changed to use the MS TrueType *Wingdings* font as it contained most of the basic geometric shapes required. Among the other noticeable changes were the addition of an *aura* (see Section 2.1) and the ability to play multiple streams of audio simultaneously and spatialised in stereo or 3-D audio, according to their position relative to the cursor. This was used to exploit Cherry's "cocktail party" effect (see Section 2.1) in conjunction with the *aura* mechanism (Benford and Greenhalgh, 1997). It provided a new departure in interfaces for direct sonification applications. The *aura* exploits the user's auditory range of perception to allow the user to *switch at will* as to what sound is being focused (Wickens and Hollands, 2000). In this version of the Sonic Browser the aura metaphor is used in the interface to allow for browsing of audio by playing all the icons under a circle cursor. This cursor is controlled by the user and may have its size increased or decreased while any sounds under the aura are played. It is the aura, which indicates the user's range of perception and allows for the broadening or zooming of that interest by increasing or decreasing the aura's size. The Sonic Browser application used interactive aura playback of multiple stream audio as an interface mechanism. This version of the Sonic Browser was used to investigate the combination of the visual and auditory modalities as well as investigating the 3D representation of spatial audio within the Sonic Browser interface. Exploratory studies, which compared 3D representation and stereo amplitude panning of the sounds, discovered that the users found a tighter coupling when using stereo amplitude panning. This was surprising, but it relates to the difficulties which users had in coupling a 3D audio representation to a 2D visual representation as opposes to coupling a 2D audio representation to a 2D visual representation.

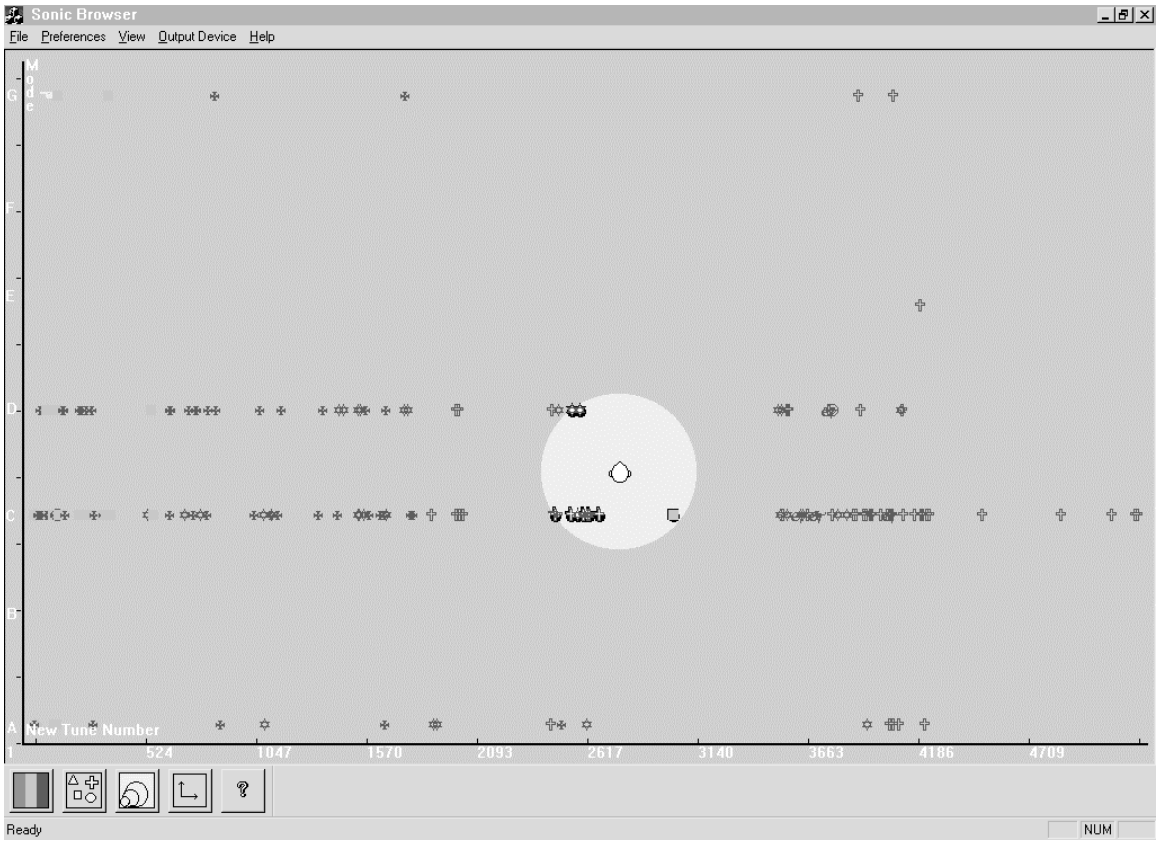


Figure 26: The Sonic Browser, Version 1

Advantages	Disadvantages
As with BROWSE system Added an *aura* and the ability to play multiple streams of audio	Application stability and performance No record structure for elements within the dataset

Table 25: Advantages and Disadvantages of the Sonic Browser, Version 1

The second version of the Sonic Browser added more robust spatialised sound (see Section 2.4.5) using MS DirectX audio libraries to the application. The use of the MS DirectX audio libraries reduced code complexity as the previous sound spatialisation was achieved through computationally expensive low-level C++ code. The audio libraries had an additional benefit of providing a more reliable system than the previous Sonic Browser by using a higher level of code abstraction based on the MS DirectX audio libraries. This allowed for users to hear sounds panned out to the left or right stereo depending upon their position under the *aura*. The version of the *Sonic Browser* as with

the previous version used multiple stream stereo-spatialised audio controlled by cursor/aura-over-icons, representing sound files. Using the Sonic Browser, properties of the sonic objects could be mapped to arbitrary features of the visual display (see Figure 27). This mapping was completely arbitrary and could be changed at any point by the user. These mappings were again mainly derived from the mappings used in the BROWSE system and the prior version of the Sonic Browser. The ability to dynamically change this mapping was a new addition to this version of the Sonic Browser. As an example, file size can be mapped to size of visual symbols, sampling rates to colour, symbol shape to file type and horizontal and vertical location to date and time. As with the previous versions the user can configure all aspects of the representation. This was implemented as a possible area of more detailed future exploration and as an interesting piece of functionality. A simple record structure for the elements within the dataset was the other addition to this version of the browser. At this point the uses of the system had grown beyond the initial scenario for use as a musicologist's search tool to a wider scope as a browsing and management tool for multimedia authoring (Fernström and Brazil, 2001). It was this idea of multimedia authoring, when further explored, which lead to the Sound Designer scenario used in this thesis. The application at this point had reached the maximum potential of its existing design due to the continued iterative development, its source code base had become unmanageable and its visualisation inherently mingled with the data management code. The only option for application in this state was to start a new code base from scratch exploiting the best design elements of the previous version.

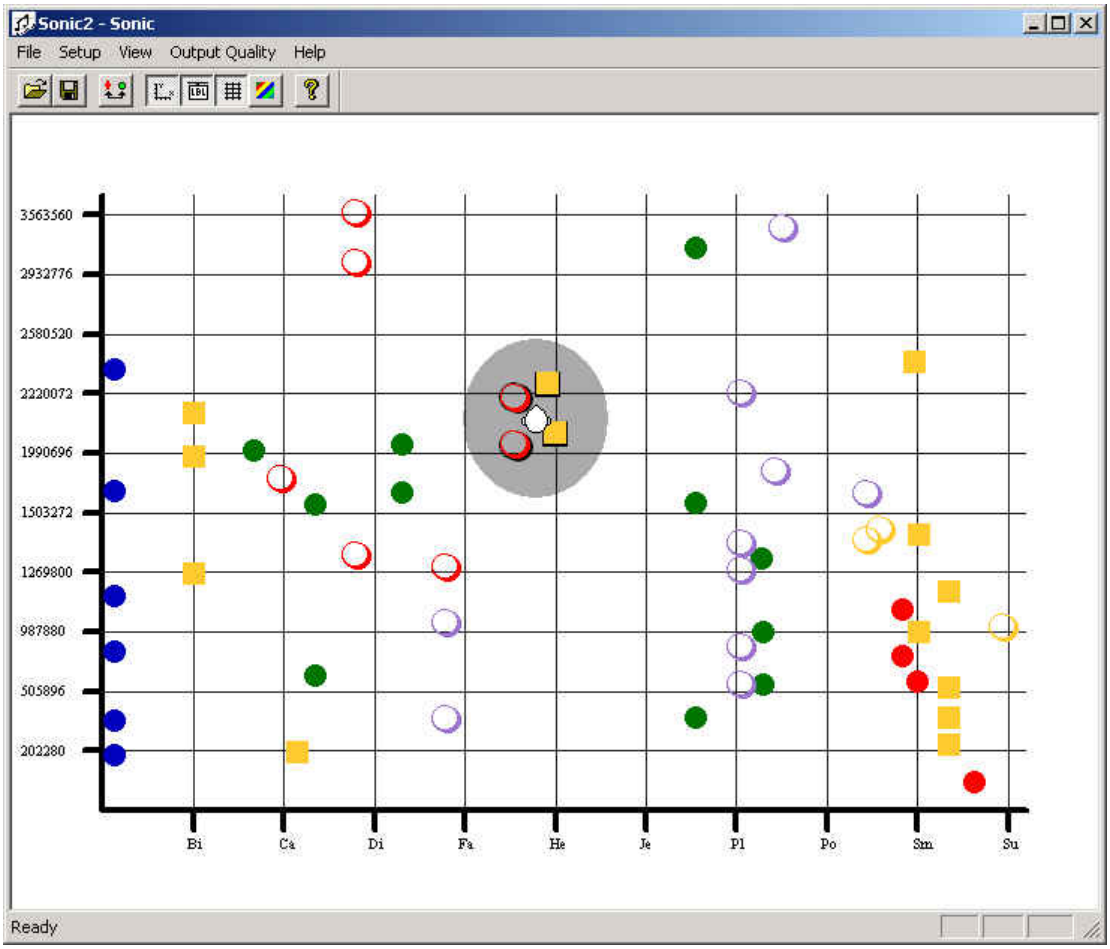


Figure 27: The Sonic Browser, Version 2

Advantages	Disadvantages
Multistream stereo-spatialised sounds	Single "star-field" visualisation
Dynamic mapping of vertical and horizontal axes according to object properties	No filtering mechanisms or dynamic query mechanisms
Record browser for objects	Limited to Windows Platform

Table 26: Advantages and Disadvantages of the Sonic Browser, Version 2

3.7 Summary of Audio Browsing Systems

The previous sections have covered a diverse spectrum of audio players, retrieval and browsing tools / interfaces in addition to offering a history of the development of the previous versions of the *Sonic Browser*. Examining the previous tools we find the standard player interface with stop, play, pause, etc functionality as the standard interface metaphor. This type of interface, due to its conceptual design, can only play a single audio resource at any one time. The Browse system and the later Sonic Browser system have a different conceptual design, which allows for the multiple simultaneous playback of several audio resources while browsing a sound collection. A standard player interface is useful for listening to audio but for browsing it may not be the most suitable interface. Our hypothesis is that when the goal is to explore as many sounds as possible that an interface with multiple audio resources playing concurrently, such as the Sonic Browser, is more suitable. Only a handful of the previous interfaces use real direct manipulation interfaces. Querying and filtering, where available, is often close to, but not quite dynamic, which is another flaw.

The various interfaces have particular advantages and disadvantages, which influenced our design. The *Microsoft Explorer* interface provides a familiar interface to many users but offers a similar presentation of file objects whether they are sounds or text documents and does not provide specific functionality tailored to the specific attributes of sound files. Standard music players such as the *Nullsoft Winamp* player offer a familiar player interface but suffers from a traditional informational retrieval design perspective where the ability of humans to perceive and differentiate multiple simultaneous streams is either unknown or unimplemented. Similar arguments could be made against the *Ninja Web Jukebox* but in its favour it does realise that sounds contain many attributes, which can be processed algorithmically for comparisons. The *SoundFisher* offers excellent algorithmic comparison abilities but fails to provide an interface differing from the standard player interface, which can equally be said of the *Marsyas* system. These systems all implement a traditional player interface. This type of implementation arose originally from a constraint where computer sound devices could only play a single sound but with new sound devices overcoming this problem a new interface metaphor is required. It also exploited an existing interface metaphor for

playing sounds, the tape recorder. Our prototype system offers an alternative interface metaphor. Our interface metaphor supports the human abilities in sound perception and segregation.

The research in this thesis can be seen to follow on from interfaces such as the *SoundFisher* interface taking advantages of recent advances in information visualisation techniques for displaying large datasets. These methods however must be taken in context of human abilities to provide an interface, which is both interactive and engaging so as to support the user whilst browsing for sound resources. Multiple views offer users something that none of the previous interfaces could, that is the ability to find relationships within the data based on different perspectives of the dataset. Dynamic queries allow a user to browse a dataset in a task domain using direct manipulation with immediate feedback of the query linked to a consistent display of the query results.

Building upon the knowledge and lessons learned in implementing the previous versions of the *Sonic Browser*, our prototype system does not seek to play sounds better than a player interface or categorise and classify them better than a dedicated classification engine but offers a new interface which exploits dynamic querying and multiple visualisations of the audio resource collection. Dynamic queries & filtering in conjunction with direct manipulation combined with direct sonification are taken advantage of to create a more natural and intuitive interface while the multiple perspectives of the data given by the different visualisations allow for discovery of new relationships in the dataset. In trying to achieve its goals the *Sonic Browser* takes a little from several aspects building upon previous work to provide an application framework, which can be used for further investigations and explorations in the area of sound browsing

3.8 Chapter Summary

This chapter has presented followed with a review of audio browsing systems from the area of sonic browsing, these are systems comparable to the focus of this research. Each system was presented in overview, outlining its particular solution to the problem and listing some of the particular solution's advantages and disadvantages. Illuminating the reasoning and the developments of previous work into our research, we presented the genealogy of the current Sonic Browser and traced its development from initial prototype to the starting point of this research. We have outlined the differences between our research approach for our prototype system and both previous *Sonic Browser* versions as well to other audio browsing systems.

The next chapter discusses the different aspects involved in the creation of our prototype system and elaborates upon the design decisions taken for several of the prominent subsystems. A brief collection of the problems encounters and a discussion of the insights gained from the implementation of the design concludes the chapter.

Chapter 4

Implementation

4 Implementation

In the previous chapters a variety of different systems for browsing and management of audio collections were presented. Although for presentation purposes they were described separately, most share similar characteristics and basic design principles. This chapter deals with our prototype system, which is a browsing and management tool for audio collections. Suggestions from users who participated in previous evaluations of the Sonic Browser expressed a desire for an application that would not suffer from portability issues across platforms. Several of the users liked the Sonic Browser but were unable to use it on other platforms such as Linux or Macintosh operating systems, which lead to the desire for portability of the application. After examining various implementation options the choice of implementing our prototype system in the Java programming environment was selected. This ensures cross platform portability as long as the Java runtime environment is installed. An early design decision was to implement the application code base from scratch as the previous versions of the *Sonic Browser* code base offered very little code to be directly reused. This portability, as previously stated, is not a central theme within the research for this thesis but is a desirable property of our system.

Amongst other objectives was the ability for the application to hold persistent data that logged any changes made by users within the application. The creation of a client-server framework was implemented to reduce computational demands on the client computer. Another advantage to implementing a client-server framework regards resource sharing. Taking a sound designer as our typical user, we find that they may work in a sound recording facility with several other sound designers and they share the same common audio collection. By offering a shared or common collection via a client-server framework, users can easily share and access sounds with the other users. The main objective of this work was to further research and expand upon the previous *Sonic Browser* interfaces, which had focused on a starfield display in conjunction with direct sonification, by adding multiple visualisations and dynamic queries. The choices were dictated by our requirements which included the goals of multiple visualisations and dynamic interactive querying of sound collections. Other minor requirements were the desire for portability and minimal processing on a client computer while still accessing a

central repository. The visualisations are used to offer a different view of the same data, allowing for the relationships between elements to be explored across the visualisations. The dynamic queries offer the ability to both interactively and consistently present a view of sounds across the visualisations. This objective is presented in Figure 28 as it relates to an audio collection. At the top for Figure 28 we see the coupling between the visual representations and their audio representation. The bottom of Figure 28 shows the coupling between the visualisation representations and dynamic queries. The audio collection noted at the very bottom of Figure 28 is shown as the dataset which our prototype is using.

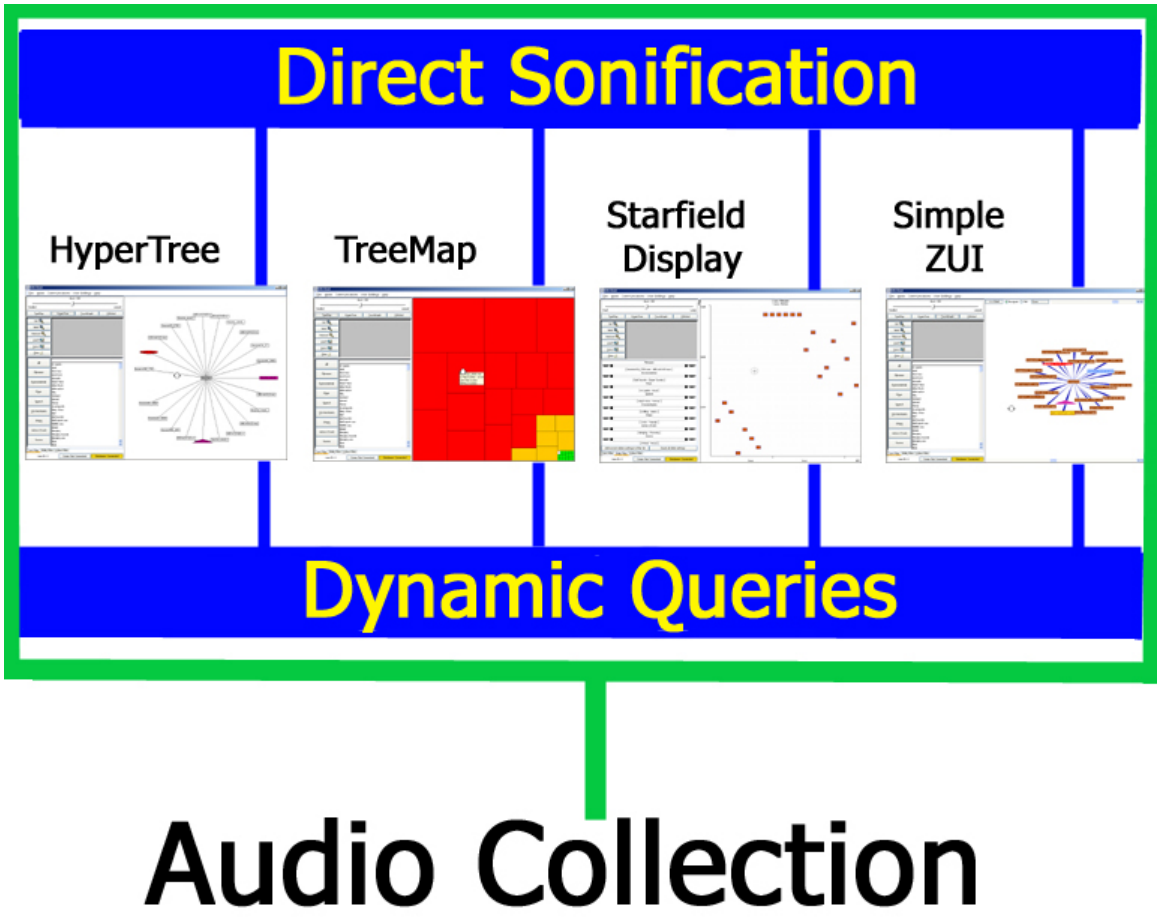


Figure 28: An overview of our prototype system

In the earlier work on the *Sonic Browser* by Fernström (see Section 3.7), the application employed a simplified interaction sequence with a tight coupling of the

interaction to create a more engaging interface. The idea behind this system was to support the human abilities for the recognition of complex and 'fuzzy' patterns through sight and hearing. The combination of the interaction sequence and the tight coupling with a direct manipulation interface was called "direct sonification". The research on the multiple views of data and the work on dynamic queries stemmed from the work on "direct sonification". Figure 28, shows how these views are operating on the same dataset, they just provide different perspectives of the data.

Our prototype system can be divided into two main areas, visualisation and sonification. The differences between the earlier *Sonic Browser* research (Fernström and McNamara, 1998) and this work is the focus on a portable, distributed graphical user interface with multiple visualisation components complemented by dynamic query mechanisms. The idea is to present different perspectives, which offers new possibilities for exploration and discovery of previously hidden relationships within an audio collection and dynamic queries and to investigate the use of our prototype system within a real world situation and examine the users experiences with our system.

The distributed element of this system applies to the client – server paradigm in network computing (see section 4.1). This differs from the earlier system, which was a standalone system, and by leveraging much of the processing onto the server; the client reduces much complex processing from the users computer by performing the calculations on the server. This complex processing consists mainly of the classification, storage and manipulation processes within the system allowing the client to provide presentation and interactivity processes. The client aspect of the system allows for the use of a central repository of sound resources and allow many users to access the same resources, which reduces the burden of managing a large number of individual sound collections. The server acts as a distributed repository for users' sound resources. Sound collections can be on the same computer, or in the same local area network (LAN) of connected computers or anywhere a user's computer can connect to the server. Portability is a problem when an application must be distributed to multiple different platforms. The previous versions of the Sonic Browser were limited to the Microsoft Windows operating systems. As part of the new system an initial requirement was to maximise cross-platform portability, which was achieved through the use of the Java programming environment.

This cross-platform portability has allowed for the testing and running of the new system on Windows, Linux and Macintosh operating systems. While these are important changes from the previous versions of the Sonic Browser and a large section of the prototype system, they are not the focus of this thesis but help to provide a stable platform for further research.

The use of multiple visualisations is an effort to support exploration of relationships across the resource collection by allowing for a collection to be viewed using several different visualisation techniques. This allows for the discovery of otherwise hidden relationships within the data. Our implementation of dynamic queries within the prototype system allows for the selection of areas of interest by removing unrelated data from the display. The different query mechanisms allow for the interactive filtering of data by selecting criteria based on user-defined properties of the data. Before defining the partial user-defined properties, we must first present an introduction into what kind of properties would be useful for classifying audio resources.

The categories we used were prompted by the previous classification schemes already mention (see Chapter 2) and we decided that user-defined properties in our prototype system for an audio resource would be Filename, Environmental, Music, Speech, Onomatopoeia, Action/Event and Source. Shape and colour properties are also user definable and can be assigned to any properties of the object in terms of shape or colour, e.g. an object representing a music file could have its musical key linked to the colour and its orchestration to the shape of the visual representation. It is the twin aspects of dynamic queries and multiple different visualisations of a sound collection, which are the focus of this thesis.

This chapter is mainly about architectural design and implementation issues and is useful for readers who are interested in implementing practical, real-time digital audio browsing systems.

4.1 Architecture

The prototype system derives most of its functionality, particularly its portability, from Sun's Java language and runtime environment. Support for distributed object programming comes from the Remote Method Invocation (RMI) (Sun Microsystems, 2002), which is discussed in greater detail in Appendix A. Our prototype system uses a client-server architecture with both elements currently written in Java. The server contains the database and network management. The client contains only the user interface and communicates with the server via remote procedure calls. This breakdown has the advantage of decoupling the interface from the management code and allows different interfaces to be built that use the same underlying server functionality. For example, the server can be accessed by different graphical user interfaces, scripting tools, web crawlers, etc. Another advantage is that the server or the client can be rewritten in some other programming language without needing to change the other part as long as the interface conventions between the two are respected. The separation of these aspects has shown great gains over previous versions of the *Sonic Browser*, especially in the interface and database aspects that now bear only a superficial resemblance to previous implementations. A more detailed view of the architecture and styles used in our prototype system can be found in Appendix B.

4.2 Development Environment

The prototype system was developed using the tools and applications:

- Java 2 SDK version 1.4
- Eclipse Open Source Java Integrated Development Environment
- Windows 2000 Professional with Service Pack 3
- MySQL Database Server Version 3.23.53

Many of the tools used are freely available on the Internet and both Eclipse and MySQL are open-source projects.

4.3 Implementation Overview

In this section a brief overview of the implementation of our prototype system is presented followed by a discussion of the lessons learned in building and testing the system. A simplified diagram of our prototype areas of functionality is shown in Figure 29. It highlights the different areas and divisions of functionality within the application but it is not a detailed view of the application's structure. The three main areas of importance to the structure of the application are the presentation / GUI implementation, the database implementation and the communication implementation. The Client aspect is shown on the top of Figure 29 and deals with the user interface elements and the sound playback. The Server aspect handles the queries, audio collection management and the database as well as returning the results of queries to the client and it is shown in the bottom of Figure 29. The two arrows in Figure 29 illustrate the application flow from the client to the server where queries are passed from the client to the server and then the results are passed back.

The presentation / GUI component was implemented using Java and its Swing graphical library components. The user interface uses Model-View-Controller design pattern (Friedrichs et al., 1999) to allow multiple views of the same underlying data to be accessed and modified from different visualisations. This model of the data is critical for the consistency in presentation across the various visualisations while still allowing for the use of dynamic queries, which also maintain this consistency across the visualisations

The database component was implemented using Java and its Java Database Connectivity library (JDBC). The database is used to store information about changes to the properties of audio resources such as any changes in categorisation, colour, shape, location, etc. It is this component in conjunction with the Extended Markup Language (XML) information in the configuration files which allow for a previous session to replicated. The configuration settings, which are stored, relate to various application resources and user preferences. The database information allows for the audio resources properties to be queried swiftly and return the relevant resources to requests from the dynamic query mechanisms within the system.

The communication component consists of client – server elements. The Client aspect deals with the user interface elements and the sound playback elements. The

Server aspect deals with the database aspects as well as handling all the communications processing. This has the advantage of decoupling the interface and allows different interfaces to be built that use the same underlying server functionality. For example, the server can be accessed by different graphical user interfaces, scripting tools, web crawlers, etc.

These three areas of presentation / GUI implementation, the database implementation and the communication implementation are dealt with in greater detail in Appendix C which also provides a description of some of the issues which were encountered which implementing the particular area.

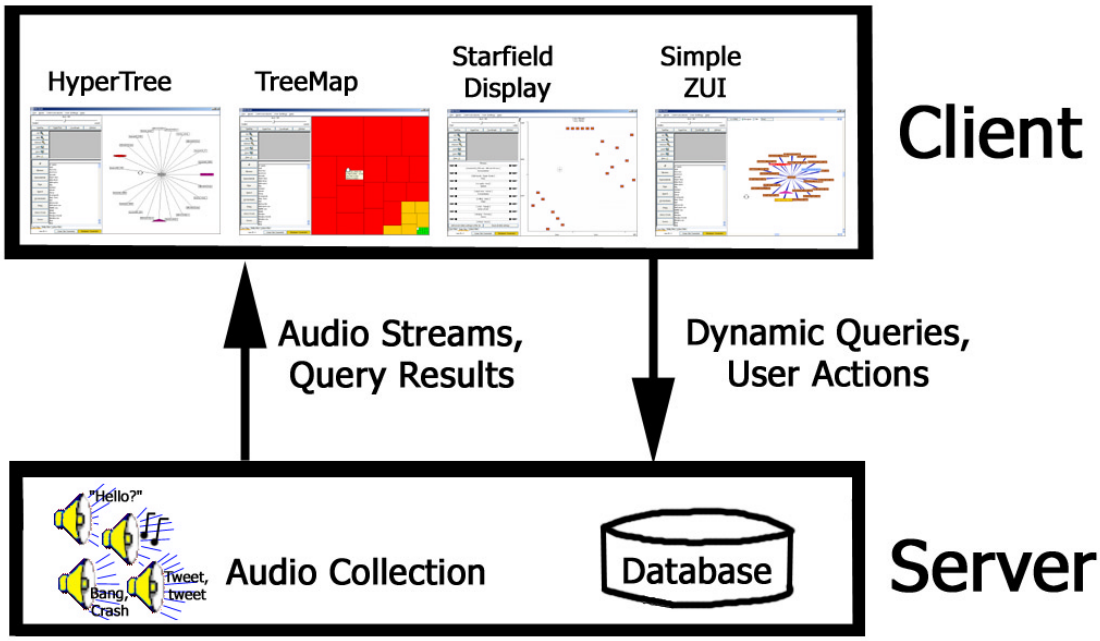


Figure 29: Simplified diagram of our prototype areas of functionality

4.4 User Interface Overview

Figure 30 shows the user interface with the different visualisations, dynamic query mechanisms and control dialogues. The user interface consists of the different visualization components (top), and the dynamic query interfaces (bottom). This illustration serves only as a first overview; details are discussed and highlighted with relate to the figure in the following paragraphs.

The different visualisation components are shown in the top of Figure 30. These are the HyperTree (see Section 2.2.2), the TreeMap (see Section 2.2.4), the Starfield display (see Section 2.2.5) and a Zoomable Interface (see Section 2.2.6). These visualisations all operate on the same underlying model allowing for changes in any one visualisation to be immediately reflected to all the other visualisations. Each of the visualisations is shown on the right hand side of the relevant picture, the visualisations can be expanded to encompass the area on the left hand side using the buttons highlighted in Figure 31, which is occupied by the query mechanisms. An example of the Starfield display when it is expanded to its maximum size is shown in Figure 32.

The middle of Figure 30 shows a close up of the tab pane used for selecting the different dynamic query mechanisms. Each of these mechanisms is available in any of the visualisation and the arrow pointing at it in Figure 30 highlight this. The arrows pointing away from it point to a close up of the relevant dynamic query mechanism.

The bottom of Figure 30 shows the different dynamic query mechanisms. The first is the text property filters, which allows for the choice of multiple properties of the audio resources that can then be filtered using the property filter controls. The property filter controls can be seen in the middle left of each of the visualisations and in greater detail in Figure 33. There are various options available for filtering that allow for the logical Boolean operations of “ORing” and “ANDing” of object properties. These controls also control the resetting, loading, saving and clearing of existing filters. The loading and saving controls for the property filters are useful as they allow for a particular combination of property filters across the various query mechanisms to be combined and saved for future use if the combination is found to be useful. The property filter controls act across the three different dynamic query mechanisms.

The second dynamic query mechanism shown in Figure 30 is the property slider filters, which are based on the Alphaslider (see section 2.3.1) and the Lensbar (see section 2.3.4). This mechanism uses alphabetic index based on the different filter properties, one slider for each filter category. This index is provided below each particular slider to show its specific range settings. The slider ‘s thumb is then used to scrolling within the range of the current filter category’s settings. This type of slider mechanism allows the user to control both the slider thumb and the range that the thumb operates in. The last dynamic query mechanism shown in Figure 34 is that of the colour property filter.

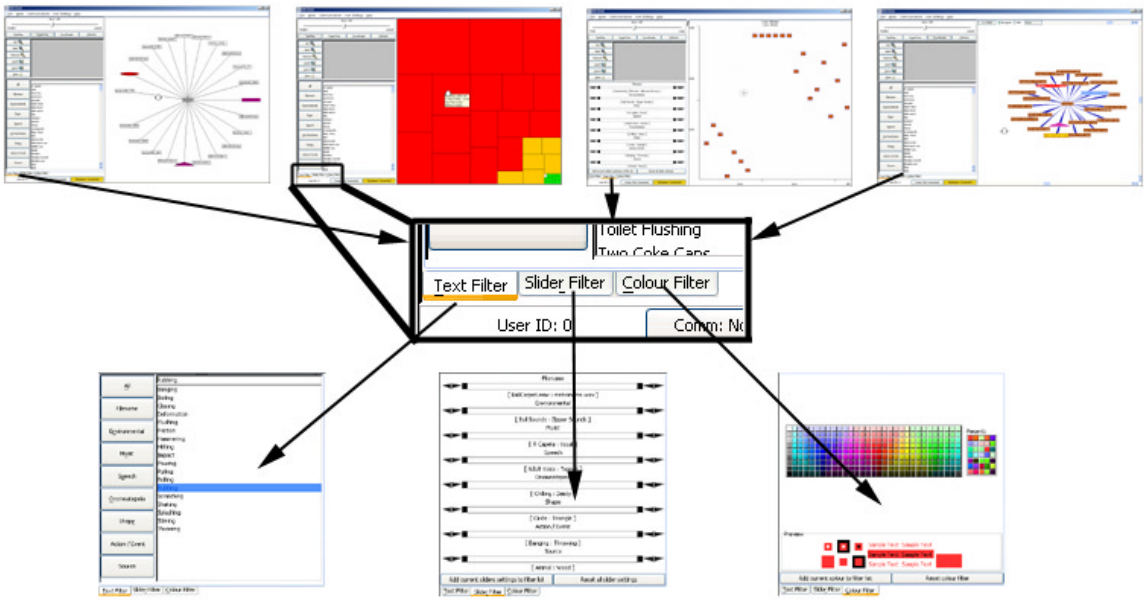


Figure 30: Overview: Our prototype system’s user interface

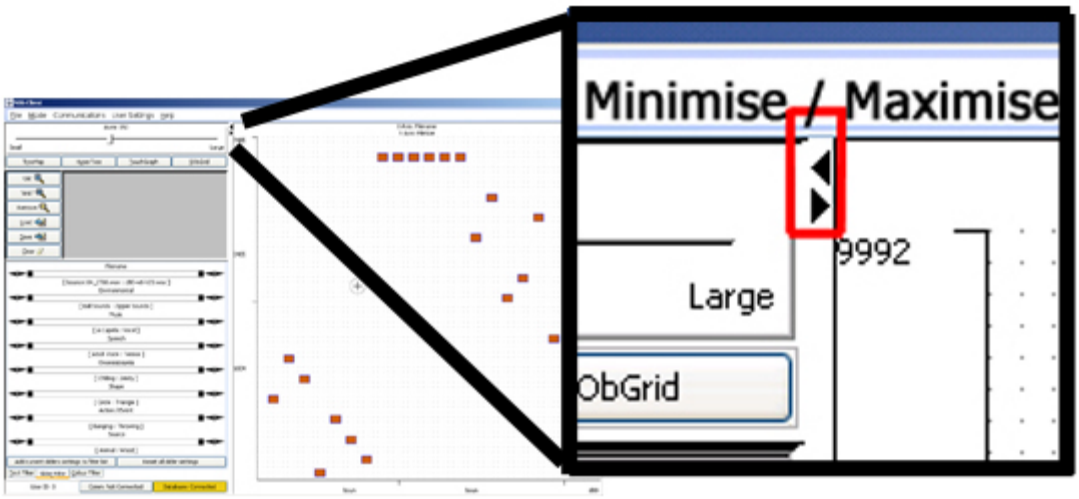


Figure 31: Expand control for visualisations

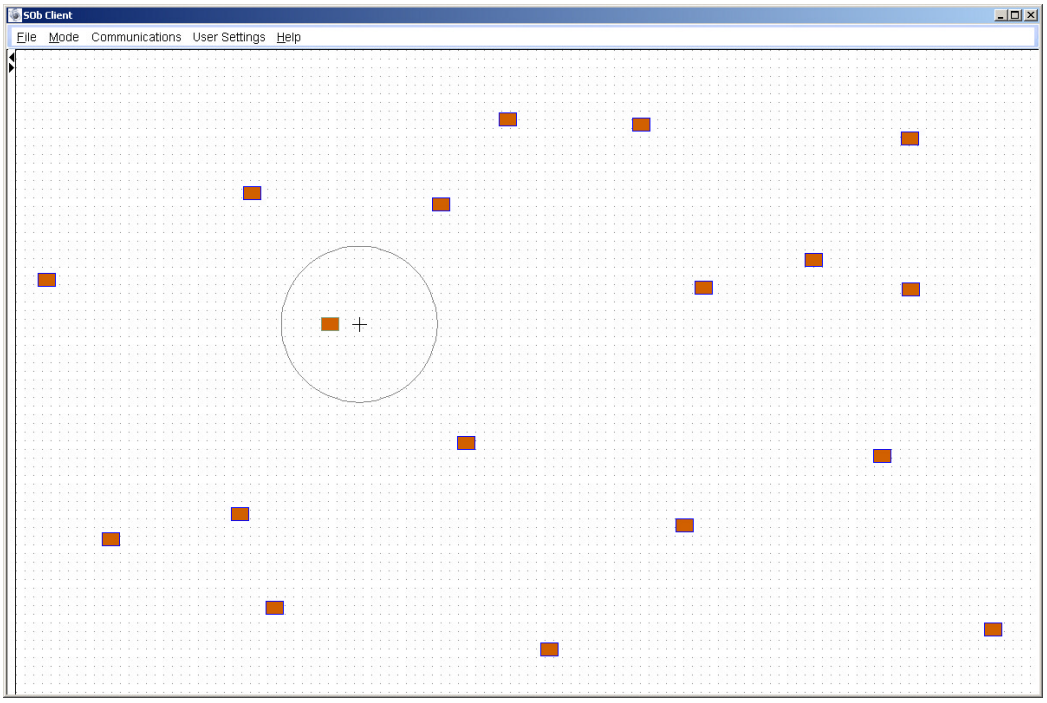


Figure 32: Starfield display visualisation expanded

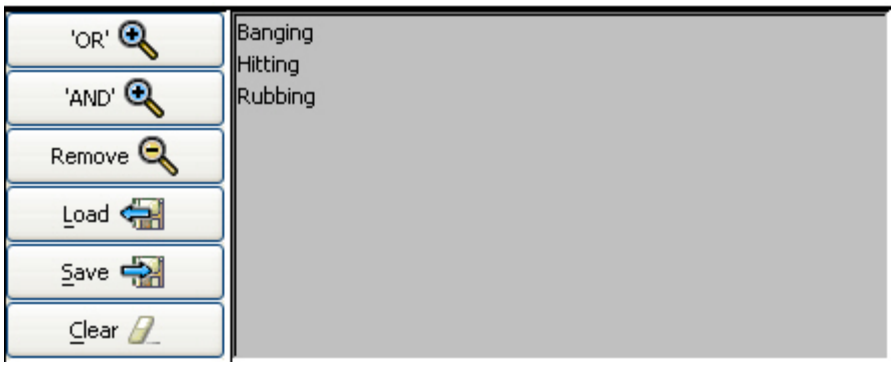


Figure 33: Property Filtering Controls

4.5 Chapter Summary

In this chapter, we have presented the architecture outline of our design, discussed the development environment and presented overviews of the application's implementation and of its user interface.

The next chapter presents the experimental procedures and techniques involved in the scenario we are conducting an initial investigation into a management/browsing of sound library collections by sound designers. The scenario is discussed in greater detail and we are presented with the experimental design of the tasks necessary to investigate the scenario.

Chapter 5

Test & Evaluation

5 Test & Evaluation

In this section our main scenario is defined and the experimental procedures and techniques elaborated. In the testing software logging was used in conjunction with the note taking, verbal protocol techniques and video analysis as it cleared up ambiguities in the interactions.

5.1 A Tool For Managing And Browsing Sound Collections

A Sound Designer is a person, who is called for the creation of special audio pieces used in radio, music, film or advertising. In the case of film, a sound designer is given a script with the audio cues highlighted. An example would be a car chase through a busy city with multiple sounds needed at various different stages of the chase to create the sonic environment to assist the visual presentation of the film. This particular instance will more than likely require the creation of a new sound sequence. The sound sequence is often created from a stock library or collection of sound components that the sound designer has at their disposal. These collections can vary in size from hundreds to several thousands of sounds, with descriptions varying from detailed to cryptic. This can lead to a slow and frustrating process while the sound designer seeks the necessary sounds that are required for the current soundscape. In larger studios there may be a designated person whose title is Sound Librarian who performs this task (Yewdall, 1999) but the Sound Designer must still choose the sounds.

There are several audio tools available for this type of work but they are generally audio authoring tools such as SoundForge on the Windows operating systems or SoundHack on the Macintosh operating systems. The few sound library tools available are database management tools. The lack of a suitable environment or tool for this type of work was one of the motivating factors in exploring this scenario. The size of sound libraries can be vast and to assist in browsing and searching for sounds there are several indexing systems used. Two common in the film industry are “Topic-Roll-Cue” and “Linear Numbering” (Yewdall, 1999). The type of naming convention that we used as

our audio file naming convention for the tasks was to use the name of the audio file as it appears on the source CDs used.

We conducted interviews with five Sound Designers to gain an insight of their needs and requirements in the context of their work. The interviews were conducted at Sonification of Hybrid Objects (SOHO) workshop held in Mestre, Italy from the 10th of June to the 21st of June 2002. The interviews were conducted at this venue due to the high availability of a cross-section of international participants active in the area of Sound Design. The interview question sheet used can be found in Appendix L. Here follows a summary of relevant points concluded from the interviews.

- Sound Designers can work on anything from musical productions/performances, to education, to sound design for various projects such as advertising or film.
- The number of sounds used at any one instant can range from one to hundreds but is mostly in the range of 6-12 concurrent sounds.
- Navigation can be supported by the use of radio dials linked to up to 100 sounds sorted alphabetically but beyond this number of sounds there are no techniques available to support navigation.
- Naming conventions vary enormously with nearly every person having their own 'system'. These systems vary from using the points within the score and referring to the files depending on the position within a score to names derived from a particular editing session where the sound was created.
- Some designers have basic databases of sounds built based on the particular project or production with the project or production as the basis of the naming convention.
- Standard file navigation techniques and applications are the techniques used by participants for browsing audio collections.
- Useful and desirable features in audio applications include zooming of the sound waveform / file and scrubbing of the sound file. Scrubbing as a navigation technique for audio files is where a moveable time-selection frame allows the user to "scrub" through the sound file along a time axis representing the sound with a playback speed dependant on the scrolling speed of the frame alone the time axis.

- There are several problems with the current lack of sound browsing tools used by all the participants. Existing tools are slow in loading audio resources. There is a distinct lack of suitable tools for live performance especially for the tasks of online selection and listening.
- The indexing of the collections was highly dependant on the sounds being created. Most sound names are created and based on what the sound does or how it relates to gesture and in conjunction to the actions that the sounds suggest.
- Mac OS and Linux are the most preferred operating systems by participants for their work.
- Creating new sounds can require very dense tracks or layers of sound elements. There is no mechanism in currently available tools to allow for a Sound Designer to hear a rough or draft version of a new sound without having first created the various sound layers in a sound authoring environment which can be very time consuming when the Sound Designer only wishes to hear how sound elements interact and sound in a rough version of the new sound.
- Current authoring / browsing environments in this domain are limited to the playback of a single audio resource at any one time using a varying degree of modality for the interaction.

After reviewing the findings of the interviews, we have gained an insight into the use requirements and typical tasks which will constitute our scenario. The scenario itself is based on the following hypothesis. Current audio authoring / browsing environments offer various degrees of drag and drop support but there is scope for improvement with the drag and drop mechanisms in these environments. Computers can be used to assist the sound designer so that instead of compiling a large paper list or taking a large number of sound files into the authoring environment, a smaller number could be selected and then imported to the authoring environment. Our prototype can offer a premix environment where a number of sounds can be simultaneously heard allows for the creation of better sound sequences. This is due to the fact that recording a sound will not always produce the desired effect. An example is the recording of a rain sound and playing it back which demonstrates this. The playback of the recorded rain sound is often found to be similar to

the sound of bacon frying rather than rain failing. Creating realistic sounds can require the layering of multiple sounds to produce a sound, the separate sound elements producing the desired sound. This layering is context dependent on the sounds used within the layering. By recording bacon frying layered with wind and bird sounds a rain sound can be perceived and a similar effect can be achieved with the sounds of rain and kitchen cutlery, which is often perceived as bacon frying. By allowing for a premix situation, a number of sounds can be layered roughly together to give an approximation of the end sound. If this layering of sounds is a good approximation, the files can be imported to an authoring environment and a more precise layered sound can be created.

In this work we build upon the previous hypothesis and assert that the interactivity and exploration of a sound collection can be improved, specifically by using an interface with multiple visualisation techniques and dynamic queries. Previous work (Fernström and Brazil, 2001) has highlighted the requirement of a selection feature, allowing for the marking of sounds of interest. This mechanism can be used to improve the interactivity and exploration of sound collections. Additional sound files can be dragged and dropped resulting in the files being added to the sound collection, which is another useful mechanism for managing sound collections. The addition of multiple perspectives on the data allows for the discovery of relationships between the data. Dynamics queries and filtering in conjunction with direct manipulation and tight coupling ensures an engaging interface. The results of this review were then used in the defining the requirements for our prototype system's functionality and in influencing the design of the exploratory investigation for this scenario.

This exploratory investigation consisted of participants being asked to browse the sound collection for a selection of sounds matching particular properties or objects. In each specific task, the participants were allowed to move the cursor around freely in the GUI trying to find target sounds as well being able to change the current visualisation to compare relationships between sounds in different visualisations.

5.2 Procedures and Techniques

There are many evaluation techniques available to the software developer as briefly outlined by Ferré (2001). Preece (2002) and Raskin (2000a) offer a broader and more illuminating coverage. In this research four techniques have been used in a qualitative evaluation:

- Note taking and verbal protocols
- Video recording
- Questionnaire and post use debriefing
- Data logging (by the application itself)

5.2.1 Note Taking And Verbal Protocols

A verbal protocol is usually gathered as part of the audio record from a video recording but it may be recorded separately. The *Thinking-Aloud Protocol* (Gould, 1988, Lewis, 1982) is one of the best known verbal protocols and makes users externalise their thoughts whilst performing tasks. This can be a useful method for obtaining 'soft' feedback from the user, focusing on the acceptance and understanding of different aspects of a system. This type of feedback and level of comments can prove very useful in understanding the user's perception of a system and the problems encountered while using it. Boren offers several practical points and illuminates some common problems, which should be taken into account when designing and running experiments using the *Thinking-Aloud Protocol* and other verbal protocols (Boren and Ramey, 2000). These points offer useful tips including on how to set the scene, how to deal with contingencies and techniques for proactively eliciting additional information. According to a review of papers reporting on studies using *Thinking-Aloud* by Nielsen et al (2002) p102 "*it is often stated that the think aloud technique applied is different, which seems to mean different from the classical think aloud. Most of the authors do not discuss and detail what they did, nor reflect on the technique or any modifications they made to it*". The result of this is a cluster of related techniques, which all adopt the same title but can vary dramatically in implementation and intent.

Our research uses a simple note taking and verbal protocol approach as the classic *Thinking-Aloud* method has a weakness when dealing with capturing auditory tasks as the

user is both talking and listening. However, we did attempt to elicit comments from the participants as they were performing the tasks. In the scenario used, participants tended to comment aloud when they performed general operations. In comparison to audio and work that involves a high level of auditory information, most users tend to be quiet which is the main problem for applying the *Think-Aloud* protocol in this particular type of investigation. This is similar to previous findings by Fernström (1998) when applying this method for a similar system.

5.2.2 Video Recording

We used video recording of the participants actions as they appeared on the monitor. There are several methods for analysis of video footage with varying levels of details being examined. These are generally classified under the heading of video analysis (Mackay, 2002). Our investigations were a coarse grain level of analysis of the video to help form a better view of how the particular task was accomplished and to differences in participant behaviour as well as highlight difficulties with the tasks and the system. The method used on the video material was to simply to record the events occurring by noting their event time (on the video), the actual event and any computer action or participant speech occurring at that point in the video. These parameters, when used in conjunction with the data logging, helped to provide a clear image of the user actions for the tasks that could then be recreated for further examination. Quantitative data analysis of the video was limited to simple measures of task completion and success.

5.2.3 Questionnaire

We used a post use questionnaire to gauge participant views on several aspects of the application and the scenario. The questionnaires inquired about application usability, difficulty in interpreting the layout of the application, sound playback delay and dynamic query mechanisms among other questions. Each question used a 7-point Likert scale. The Likert scale was chosen to allow of the participants strength of agreement to be measured with a clear question. The questions used semantic differential so that the scale has bi-polar adjectives (such as easy-difficult, clear-confusing) at the end points of the scale. The use of an odd point scale was to allow for a clear centre point. The questionnaire was

based on a questionnaire developed by Constantine et al (1999). The questionnaires used in the experiments can be found in Appendix H.

5.2.4 Data Logging

The *Sonic Browser* performs application specific interactive logging when any changes to the properties of an object occur. These are recorded with the time of the change or action to a database to allow for a later analysis. This logging is made explicitly clear to the user in the consent form and in the initial introduction to the *Sonic Browser*. The data logging was reiterated to the participants by making them explicitly choose the menu option to save a user report file at the end of their session. The user report file is shown to the user after the tasks have been completed and can be used in conjunction with analytical tools such as Math lab and SPSS. This detailed analysis uses techniques such as object clustering to illuminate further details of a particular user's session data as well as combining several users sessions to gain a broader perspective to many questions for instance, object classification. A sample user report file can be found in Appendix I.

5.2.5 Participants & Experiment Design

The participants were 15 University of Limerick postgraduate students.

In this scenario, a within-subject experimental design was used, i.e., each participant carried out the tasks on three different visualisations, the HyperTree, the Starfield display and the Zoomable interface our prototype system. The TreeMap interface was excluded from the testing due to resource and timing demands as well as issues regarding its suitability for this type of task. In order to avoid sequence effects, task order and interface order were counter-balanced within subjects.

5.2.6 Experiment Equipment

The computer used in the experiments was a dual Athlon 1.6 MHz RAM 1 Gigabyte with a 19-inch computer monitor, a Sound Blaster Live! Platinum 5.1 sound card and a GeForce 3 64MB graphics card running Windows 2000 with service pack 3. The Java 1.4 release 1 runtime was used with MySQL version 3.23.53. The sounds were presented to the participants using stereo headphones. The evaluation environment is shown in Figure 34.

Figure 34: Evaluation environment

5.2.7 Procedure

All the participants in these experiments filled out a consent form as shown in Appendix J as well as being given a short introduction to the *Sonic Browser* before beginning any of the tasks. A session lasted no longer than one hour and consisted of three stages for both experiments and an additional practice task for the tool for managing and browsing sound collections scenario.

1) Introduction and Training: The observer set up the appropriate interface and briefly explained the interface to the participant. The participant was invited to try-out the system and get comfortable with it. Participants were given unlimited time to try-out using the system and were free to ask any questions. Actual practice time varied from two to ten minutes for each participant. The query and filtering mechanisms required significantly more training than any of the other visual interfaces.
2) Practise task: A practice task (similar to the complex query question used) was given. During this task, participants were free to ask questions about both the task and the interface.
3) Timed tasks: The task list was given on paper. Participants read each question and were asked if they fully understood it. This was done to eliminate variations in subject comprehension speed. When the participant was ready, the observer started videoing and note taking. When participants found the answer, they expressed it verbally. The various task sheets can be found in Appendix K.
4) Subjective evaluation: Participants were asked to fill-out a questionnaire after having completed the tasks. 18 questions were for the sound library scenario and are shown in Appendix H.

5.3 Chapter Summary

This chapter has presented a review of experimental procedures and techniques used for investigating our research. A description of the experimental design and environment for the scenario to be investigated is presented, followed by a discussion on the scenario.

In the next chapter, we present the results of using our prototype system in an exploratory investigation. The exploratory investigation was an investigation of the tool for managing and browsing sound collections scenario.

Chapter 6

Results & Discussion

6 Results & Discussion

This chapter deals with the results of testing our prototype system in the scenario specified and then offers a summary of the results of our investigation followed by a discussion of these results.

6.1 A Tool For Managing And Browsing Sound Collections Results

This section presents the results of the participants browsing results where they sought a selection of sounds matching a particular property or object within the sound collection. A brief summary of the results is presented. This research is interested in the suitability and engagement of our system. As discussed in the previous chapter, the sound collection was presented using stereo headphones and using our prototype system. During experiments the participant's sessions were videotaped.

6.1.1 Results of object tagging for particular properties or objects

The recorded sounds used in this experiment were drawn from nine sources, seven commercial sound effects CD's, a local collection of everyday sounds and a collection of synthesised sounds of bouncing balls from the SOb project. The length of the sounds varied from 0.1 to 50 seconds.

In general, all participants carried out the tasks successfully. There were three particular circumstances of where this was not the case.

- Selected a single sound with the desired property/properties when they were looking for any sounds with the desired property/properties. The result of this task was that one of several sounds was found and tagged but other sounds present, which fulfilled the criteria, were not selected. This could simple be a misinterpretation of the task at hand.
- No sounds matching the description were found and the participant simply saved a blank file without any tagged files.

- One case showed that at times participants might not clear all previously tagged files, which causes additional incorrect data to be saved as sounds fulfilling the task criteria.

6.1.2 Results of Questionnaire

At the end of each session as part of the participant debriefing, a questionnaire was presented to the participants in order to gain an insight into their feelings about our prototype system. The participants responses to our system and to the tasks in the scenario were examined by filled out a seven Point Likert scale questionnaire using the questions shown in Appendix H with six sets of semantic differentials. (From 0 to 6, where 0 is ``negative'' and 6 is ``positive''). The reasoning and clarification of this choice was presented in Section 5.2.3. The details and results are provided in greater detail in the Appendixes, see Appendix H for the questionnaire and Appendix M for the detailed results of the questionnaire.

Questions one, two and seven dealt with interpretation, learnability and ease of use of our prototype system. The results of these questions show that the users' find our prototype system easy to learn and use. Questions three to six dealt with the filtering mechanisms of our prototype system. The results of these questions, confirmed by video footage, illuminate several items such as it is always easier to find a sound when you know its filename and that the filtering mechanisms were found to be easy to use. Questions eight concerned a slight delay when playing an audio file with our prototype system. The results of this question show that in our scenario this delay was not appreciable and did not affect the task. This allows us to say that for tasks in our scenario that while audio playback delay this is still an important factor a greater play delay is acceptable. Questions nine and ten deal with realism and quality of the sounds, which were found to be excellent by participants.

The essence of the results of this questionnaire show that users find our prototype system easy to use and easy to locate sounds with as well as above average for performing queries on the dataset.

6.1.3 Summary of Results for Sound Library Experiment

The rich verbal protocol returned several interesting results during the experiment. The play delay was only highlighted by one participant who ``expected sound to be instantaneous, not take ages". This was more likely related to one of three causes either silence at the start of the sound or a sound beginning with a very low volume or due to a technical issue with using Java on specific operating systems. The technical issue with using Java may result in the presence of a sound playback delay, which may be due to the Java Media Framework architecture, as the addition of an extra layer for the Java virtual machine above the operating system could add a slight delay to the sound playback on the Windows platform. The slight delay was encountered intermittently during the testing of the application on the Windows platform and may also be due to computational demands or possibly due to the Advanced Micro Devices (AMD) architecture that was used as the testing platform hardware. The AMD architecture is a rival but compatible standard to the Intel architecture. Application testing on the new Macintosh OS X system (Linux based OS) did not result in this type of sound delay. The source of the delay should be examined in further testing related to the addition of a layer of communications which Java must use to communicate to the sound devices when using the Windows operating system and is a possible area of research to determine Java's validity as an environment for real-time multiple audio stream systems.

A unique browsing behaviour was used by one participant, who used the text filtering mechanisms and then removed these filters. This behaviour allowed the user focus on objects of interest and then return an overview of the dataset, this allowed for a point/s of reference to be determined prior to browsing the sounds within the visualisation. This type of behaviour indicates the usefulness of implementing dynamic query mechanisms as they allowed the user to get a better overview of the collection using waypoints / reference points. The HyperTree visualisation was commented on favourably by most of the participants. They liked the layout and novel visualisation technique as well as the labelling of the nodes "Here I can see the file name so..." which assisted their navigation within the collection. The zooming scrollbar in the TouchGraph visualisation was found to be both difficult to use and to see by several of the participants who commented such.

The use of an aura for rough layering of sounds was confirmed by the participants comments especially the statement by one participant that the “aura allows for quick checking of groups for particular sound’s which is useful due to playing all sounds in the aura at once”. This behaviour was also supported by drag and drop within the application, which some participants used to move sounds into groups of sounds. These groups contained sounds, which were similar to the criteria and were later refined to eliminate those that did not meet the criteria. This use of the aura as a premix environment supported by dynamic queries and multiple representations of the collection was successful in this scenario but requires further investigation.

Other issues were discovered in the debriefing and through user comments during testing, mostly relate to future improvements of our prototype system such as the ``Text lookahead should look for `Flushing toilet' as well as `Toilet flushing' '' and the addition of a ``Right click context menu with options for setting object properties and for group of objects properties".

6.2 Summary of Results

- The management of a sound collection is a difficult task but it can be made easier through the use of dynamic querying combined with both direct manipulation and direct sonification of the sounds within the collection.
- Multiple visual representations of a collection offer an engaging method of exploration of the data allowing for the discovery of previously unknown relationships between data.
- An aura offers a useful premix environment for hearing a rough layering of the selected sounds prior to editing a new sound in an audio authoring environment.
- In our particular scenario we found that minor temporal delays in playback are not evident to the participant when they are selecting the necessary sounds to meet a task's criteria.

From the participant's comments in conjunction with video footage, we find that users remember where they heard a sound before, hence users that browse with the *aura* hear more sounds, and will sometimes be able to recall the spatial distribution of sounds. This indication is also supported by previous work by Fernström (1998) and also by Kobayashi and Schmandt's results (1997).

6.3 Discussion

The results from the evaluation offer several interesting questions as well as illuminating new answers.

The results of the sound library experiment show that an overview with dynamic querying does improve performance for browsing tasks, but it can sometimes lead to ambiguity as users' classifications of sounds may differ. From our results, we found that when a sound collection uses arbitrary classifications such as descriptive file names, specific sound sources or sound events classifications that browsing will be faster unless classification problems arise. We also found that a hyperbolic layout for browsing makes it both easy and enjoyable for a participant to explore a sound data set. Exploring sound collections is difficult but it can be made easier through the use of dynamic querying combined with both direct manipulation and direct sonification of the sounds in the collection. Multiple representations of the collection allowed for an engaging exploration and allowed for the discovery of previously unknown relationships in the collection. Our prototype system was found to be an engaging and interactive interface, which provided functionality that satisfied requirements. The requirements sought the ability to select or highlight sounds of interest, the ability to export these sounds to an authoring environment, the ability to drag and drop the sounds within the environment, the ability to import new files into the environment and most importantly the ability to premix sound elements to create rough sounds which can then be used to create a new sound which consists of those particular elements. In the context of the sound designer scenario, by meeting our hypothesis and providing the support mechanism discussed in Chapter 5, we claim that our prototype system fulfilled the requirements for the scenario.

The scenario, while offering useful insights into the domain, was exploratory and did not return the depth of result needed for more detailed results and metrics. This scenario should be further investigated using more formal evaluation techniques. The suggestive results from our investigation have proved the general suitability of our approach for use in the sound designer scenario. It did not, however, return the depth of result or empirical comparisons that could be used in comparing our system with other systems or approaches to the particular scenario. The use of "discount" evaluation techniques did not give us the depth of results required for these types of comparisons so

in retrospect more rigorous methods should be used and offer an area of future investigation.

The scenario offers further possibilities for more detailed experimentation and also suggests new requirements or functionality for inclusion in future versions. Notable suggestions include a right click context menu via the mouse, audio scrubbing for swifter browsing of individual sounds and the inclusion of sound spatialisation as well as existing sound panning for sound playback. These are just a few of the suggestions; a more complete review of possible future work is included in the next chapter in the section of the same name.

6.4 Chapter Summary

In this chapter, we've presented the results of the scenario where our prototype system was used to investigate two distinct scenarios. A short discussion on these results is then presented.

These results allow us to drawn conclusions about our prototype system and its suitability of the scenario as well as highlighting improvements and directions for new research. The next chapter presents a summary of the work in this thesis, discusses our current research and possible future research directions. It concludes with some thoughts about our prototype system as a new and future research area.

Chapter 7

Summary and Conclusion

7 Summary and Conclusion

In this thesis, a new user interface for browsing audio collections using multiple visualisations, dynamic queries and direct sonification was proposed, developed, and evaluated. The system combines elements from information retrieval, sonification, browsing and information visualisation. Some of the main characteristics are: 1) ability to browse large audio collections interactively 2) multiple visualisations of the collection which allows for new discoveries within the collection 3) dynamic querying of the collection which remains consistent across the visualisations 4) subjective results of its use in explorative probes investigating a use scenario.

To summarize the main contributions of this thesis are: 1) the determination of the suitability of different visualisation mechanisms for a specific scenario 2) an application framework which can be used for further investigations and explorations into the area of sound browsing. Our prototype system enabled users to employ dynamic queries, direct manipulation and direct sonification across multiple visualisations to pose queries by direct manipulation. It reduced a difficult cognitive task for sound designers into a simpler perceptual task by allowing them to dynamically query and select desire sounds. As the number of visualisation systems that employ direct manipulation techniques supported by dynamic querying is increasing, our approach for audio collection interfaces is likely to have widespread impact on future designs.

Our hope is that someday in the future, the interfaces for the browsing of audio resources and collections will be inspired by our system. In the following sections about current and future work related to this thesis will be described.

7.1 Current Work

In this section our ongoing work is briefly outlined, which continues the research described in this thesis. As already mentioned, in addition to sound collection management, which is an important area for Sonic Browsing research, we are collaborating with people involved in digital sound editing applications to use our framework and interfaces to visualise and assist in the organisation, searching and browsing of the collections used. One area of continuing collaboration is in developing

new interfaces for music information retrieval (see Section 2.4.2). The CPN View library, a class library for representing music scores, has been in development at Limerick for the past number of years (Ó Maidín, 1998, Ó Maidín, 1995). Work has been ongoing to develop a framework for music information retrieval (Ó Maidín and Fernström, 2000) and has resulted in a new application CPN Browse that combines the best features of CPN View with the Sonic Browser (Fernström and Ó Maidín, 2001). New technologies and languages such as the .Net framework (Richter, 2000) are being examined as possible solutions to one of the problems encountered with the current prototype system that of a slight play delay on playback of sounds. New music formats such as the open-source Ogg Vorbis (Moffitt, 2001) sound format are being examined for inclusion due to both there increasing popularity and the advantages of their open-source license. The advantages of an open-source license such as the one used by Ogg Vorbis prevents proprietary or patent / royalty issues arising as has happened with the MP3 format (Lam and Tan, 2001).

7.2 Future Work

Sonic Browsing is a relatively new area of research with many new and interesting possibilities for future work. Many of these new directions and aspects are obvious extensions of the work and it is our hope, that the software we developed will help us and other researchers to explore and realise some of these new possibilities.

Marsyas, as described in section 3.6.5 is an audio analysis, classification and audition framework developed at the Princeton University, USA. We are currently collaborating with the Marsyas team, combining their system with the Sonic Browser. More specifically the new system, under development, combines the interactivity and user configurability of the Sonic Browser with the automatic audio information retrieval tools of Marsyas in order to create a flexible distributed graphical user interface with multiple visualization components that provides novel ways of browsing large audio collections. The initial ideas and overview of this work were presented at the International Conference for Auditory Display (ICAD) 2002. An introduction into the Marsyas framework and many audition research topics such as segmentation and classification of audio signals was discussed by Tzanetakis (2002). Our work on combining Marsyas and the Sonic Browser has suggested the advantages of combining

automatic audio information retrieval with multiple visualisations, dynamic queries and direct sonification. This inclusion of automatic classification of audio into existing audio browsing systems is an avenue of research with many future possibilities for new sound browsing interfaces.

The investigation of new visualisations methods and techniques is an active research area. Many of the visualisations mentioned in this thesis have undergone both experimental and practical use. There have, however, been few investigations into applying these techniques in conjunction with one another to provide multiple views of the same data. By providing an example of this type of combination in a particular domain, we hope that others will apply this approach to other domains.

Another interesting aspect of future investigation are dynamic query interfaces suitable for assisting browsing of audio resources and supporting these queries with audio classification mechanisms so that they not only find selected criteria resources but also resources which have similar criteria. An approach would be to incorporate both classification and similarity metrics of the audio resources into the dynamic query mechanisms.

Investigations into the aspects of thumbnailing (see Section 2,4,3), segmentation or cue points (see Section 2.4.4) and classification (see Section 2.4.2 and systems in Sections 3.3, 3.4 and 3.5) are other possible areas of future research. A combination of the three different types of techniques would be a definite area of new exploration. Thumbnailing is the process of selecting points or areas of interest within a sound file and we have provided an introduction to this topic (Brazil, 2001). Segmentation is where an audio resource is split into scenes, different speakers or between music and speech. Classification is where an audio resource is classified as a particular type of sound such as speech, music, environment or speech and music.

7.3 Final Thoughts

There were several difficult pieces in implementing the application: the use of various visualisations for representing the sound resource, the split between client and server functionality, and the additions of a database and configuration layers. However, as a whole, creating this prototype system has been a demanding but educational experience

in building an interactive, multiple visualisation and dynamic query interface suitable for the browsing and management of audio resources.

Our research had several aims: to investigation and review of the area of audio browsing, the development and initial evaluation of our prototype distributed system which implemented multiple representations, direction sonification and dynamic query mechanisms for the browsing of audio resources and the examination of users subjective experiences with our system. We hope that our research can serve as a model for subsequent work by highlighting new suitable techniques from information visualisation, sonification and browsing guided by human computer interaction techniques and design methodologies. Our approach is relevant not only to the particular problem domain of sonic browsing, but also to the fields of information browsing as well as information retrieval as these can implement similar mechanisms to support exploration. The amount of multimedia information and resources available on personal computers and via the Internet has grown exponentially over the past decade; hence there is a growing interest in multimedia data retrieval and management. This is referred to as the data availability paradox by Woods (1999), more and more data is available, but our ability to interpret what is available has not increased. We have provided a new rich and interesting system that we hope can help us to perceive more as the amount of data increases. Today as we strive to cope with the data availability paradox we must parallel this growth of data with the creation of systems for assisting humans in interacting and exploring these vast amounts of data.

8 References:

Abdul-Fatah, I. and Majumdar, S., 1998. "Performance Comparison of Architectures for Client-Server Interactions in CORBA" In: International Conference on Distributed Computing Systems (ICDCS'98), Amsterdam, The Netherlands, 26 - 29 May, 1998. IEEE Computer Society, 2-11.

Ahlberg, C. and Shneiderman, B., 1993a. "The Alphaslider: A rapid and compact selector" In: ACM proceedings CHI '94, 1993a.

Ahlberg, C. and Shneiderman, B., 1993b. "Visual Information Seeking: Tight coupling of dynamic query filters with starfield displays" In: ACM CHI '94, Boston, MA, USA, 1993b. 313-317.

Ahlberg, C. and Shneiderman, B., 1994a. "The alphaslider: a compact and rapid selector" In: Human factors in computing systems, Boston, United States, April 24 - 28, 1994a. ACM Press, 365-371.

Ahlberg, C. and Shneiderman, B., 1994b. Visual information seeking using the FilmFinder ACM CHI '94 Conference Companion, Boston, MA, USA.

Albers, M., 1993. "Sonification and Auditory Icons in a Complex, Supervisory Control Environment" In: ACM SIGGRAPH/Multimedia `93, Workshop, Sound-Related Computation, Los Angeles, CA, USA, 1993.

Albers, M., 1994. "Auditory Cues for Complex, Supervisory Control Systems: Aiding Fault Detection and Isolation in Satellite-Ground Control" In: ACM CHI '94, Workshop, The Future of Speech and Audio in the Interface, Boston, MA, USA, 1994.

Albers, M. and Bergman, A. S., 1995. "The Audible Web: Auditory Enhancements for Mosaic" In: ACM CHI '95, Denver, CO, USA, May 7-11, 1995.

Albers, M. C., 1995. Varese - Non-speech Autitory Interfaces. Georgia Institute of Technology.

Amant, R. S., Blair, J. E., Healey, C. G., Park, S., Barry, P. and Rogers, D., 2001, "Visualization and selection in a music database: A case study", TR-2001-04 North Carolina State University Raleigh.

Arons, B., 1992. "A Review of the Cocktail Party Effect" Journal of the American Voice I/O Society, 12, 35-50.

Asahi, T., Turo, D. and Shneiderman, B., 1995. "Using Treemaps to Visualize the Analytic Hierarchy Process" Information Systems Research, 6, 4, 357-375.

Bacon, J., Moody, K., Bates, J., Hayton, R., Ma, C., McNeil, A., Seidel, O. and Spiteri, M., 2000. "Generic Support for Distributed Applications" IEEE Computer, 33, 3, 68-76.

Ballas, J. A., 1993. "Common factors in the identification of an assortment of brief everyday sounds" J. of Experimental Psychology, 19, 2, 250-267.

Barra, M., Cillo, T., De Santis, A., Petrillo, U. F., Negro, A. and Scarano, V., 2002. "Multimodal Monitoring of Web Servers" IEEE Multimedia, 32-41.

Bederson, B. B., 2001. "PhotoMesa: A Zoomable Image Browser Using Quantum Treemaps and BubbleMaps" In: UIST '01, Orlando, Florida, 2001. ACM Press, 71-80.

Bederson, B. B. and Hollan, J. D., 1994. "Pad++: A Zooming Graphical User Interface for Exploring Alternate Interface Physics" In: ACM UIST'94, 1994. 17-24.

Bederson, B. B. and McAlister, B., 1999, "Jazz: An Extensible 2D+Zooming Graphics Toolkit in Java", HCIL-99-07, CS-TR-4015, UMIACS-TR-99-24 University of Maryland

Begault, D. R., 1994 3-D Sound for Virtual Reality and Multimedia, Academic Press Inc, Cambridge, MA, USA.

Benford, S. and Greenhalgh, C., 1997. "Introducting Third Party Objects into the Spatial Model of Interaction." In: Fifth European Conference on Computer Supported Cooperative Work, Lancaster, UK, 1997. Thimbleby, H., O'Conaill, B. and Thomas, P.: Kluwer Academic Publishers, 189-204.

Boisvert, R. F., Moreira, J., Philippsen, M. and Pozo, R., 2001. "Java and Numerical Computing" IEEE Computing in Science & Engineering, 3, 2, 18-24.

Bonardi, A., 2000. "IR for Contemporary Music: What the Muscologist Needs" In: International Symposium on Music Information Retrieval, Plymouth, MA, USA, Oct 23-25, 2000.

Boren, M. T. and Ramey, J., 2000. "Thinking Aloud: Reconciling Theory and Practice" IEEE Transactions on Professional Communication, 43, 3, 261-278.

Brazil, E., 2001. "Cue Points: An Introduction" In: COST-G6 Conference on Digital Audio Effects DAFx-01, Limerick, Ireland, December 6-8, 2001. Fernström, M. and Brazil, E.: University of Limerick,

Brazil, E., Fernström, J. M., Tzanetakis, G. and Cook, P., 2002a. "Enhancing Sonic Browsing Using Audio Information Retrieval" In: International Conference on Auditory Display ICAD-02, Kyoto, Japan, July 2-5, 2002, 2002a. 132-135.

Brazil, E., Ottaviani, L. and Fernström, M., 2002b, "Final report on understanding, modelling, and implementing sounding objects: Experiments in a 2-D space using the Sonic Browser for validation and cataloguing of Sound Objects", Deliverable n.13

Bregman, A. S., 1990 Auditory Scene Analysis: The Perceptual Organization of Sound, M.I.T. Press, Cambridge, MA, USA.

Brock, D., Stroup, J. L. and Ballas, J. A., 2002. "Using an Auditory Display to Manage Attention in a Dual Task, Multiscreen Environment" In: International Conference on Auditory Display ICAD-02, Kyoto, Japan, July 2-5, 2002, 2002. 357-361.

Cannon, J. W., Floyd, W. J., Kenyon, R. and Parry, W. R., 1997, Hyperbolic Geometry, in: Flavors of Geometry, Ed, Levy, S., 59-115, Cambridge:Cambridge University Press.

Cherry, E. C., 1953. "Some Experiments on the Recognition of Speech with One and Two Ears" Journal of the Acoustical Society of America, 25, 975-979.

Cherry, E. C. and Taylor, W. K., 1954. "Some Further Experiments on the Recognition of Speech with One and Two Ears" Journal of the Acoustical Society of America, 26, 549-554.

Cohen, J., 1994, Monitoring Background Activities, in: Auditory Display: Sonification, Audification and Auditory interfaces, Ed, Kramer, G., 499-532, Reading, MA, USA:Addison-Wesley Publishing Company.

Constantine, L., L., and Lockwood, L., A.D.,, 1999 Software of Use: A practical guide to the models and methods of usage-centered design, Addison Wesley, Reading, MA.

Conversy, S., 1998. "Wind and Wave Auditory Icons for Monitoring Continuous Processes" In: CHI98 Conference on Human Factors in Computing Systems, Los Angeles, CA, USA, 18-21 April, 1998.

Fernström, J. M., 1998. After Direct Manipulation - Direct Sonification : An investigation of possibilities with multiple auditory streams to enhance browsing of multimedia data sets. MSc Dissertation, University of Limerick.

Fernström, J. M. and Bannon, L. J., 1997a. "Explorations in Sonic Browsing" In: BCS HCI '97, Bristol, UK, 1997a. Thimbleby, H., O'Conaill, B. and Thomas, P.: Springer Verlag, London, 117-132.

Fernström, J. M. and Bannon, L. J., 1997b, "Multimedia Browsing", Atlanta, GA, USA.

Fernström, J. M. and McNamara, C., 1998. "After Direct Manipulation - Direct Sonification" In: ICAD '98, Glasgow, Scotland, 1-4/11/1998, 1998. Springer-Verlag,

Fernström, M. and Brazil, E., 2001. "Sonic Browsing: an Auditory Tool for Multimedia Asset Management" In: ICAD 2002, Helsinki, Finland, July 29 - August 1, 2001. Hiipakka, J., Zacharov, N. and Takala, T.: Laboratory of Acoustics and Audio Signal Processing and the Telecommunications Software and Multimedia Laboratory, Univeristy of Helsinki, Espoo, Finland, 132-135.

Fernström, M. and Ó Maidín, D., 2001. "Computer-supported Browsing for MIR" In: ISMIR 2001, Bloomington, Indiana, USA, Octobar 15-17, 2001. Downie, J. S. and Bainbridge, D.: 9-10.

Ferré, X., Juristo, N., Helmut, W. and Constantine, L., 2001. "Usability Basics for Software Developers" IEEE Software, 22-29.

Friedrichs, J., Jubin, H., Chilvers, M., Devaux, B. and Briggs, M., 1999 Java Thin-Client Programming, IBM - International Technical Support Organization.

Furht, B. (Ed.) 2000. Handbook of Internet Computing,CRC Press, Florida Atlantic University, Boca Raton, Florida, USA.

Furnas, G., 1986. "Generalised Fisheye Views" In: ACM CHI '86, Massachusetts, USA, 1986. ACM Press,

Furnas, G., 1995. "Space-Scale Diagrams: Understanding Multiscale Interfaces" In: CHI '95, Denver, CO, USA, 1995. ACM Press,

Gamma, E., Helm, R., Johnson, R. and Vlissides, J., 1995 Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, Reading, Mass.

Gaver, W. W., 1989. "The Sonic Finder: An Interface that Uses Auditory Icons. The Use of Non-Speech Audio at the Interface" In: ACM CHI '89, Austin, Texas, USA, 1989. ACM Press,

Gaver, W. W., 1993a. "How do we hear in the world? Explorations of ecological acoustics" Ecological Psychology, 5, 4, 285 - 313.

Gaver, W. W., 1993b. "Synthesizing Auditory Icons" In: Interchi'93, 24-29 April, 1993b. ACM Press, 228-235.

Gaver, W. W., 1993c. "What in the world do we hear? An ecological approach to auditory source perception" Ecological Psychology, 5, 1, 1-29.

Gaver, W. W., 1994, Using and Creating Auditory Icons, in: Auditory Display: Sonification, Audification and Auditory interfaces, Ed, Kramer, G., 417-446, Reading, MA, USA:Addison-Wesley Publishing Company.

Gaver, W. W. and Smith, R., 1990. "Auditory icons in large-scale collaborative environments" In: INTERACT'90, 1990. al., D. D. e.: Elsevier Science Publishers B.V. (North-Holland), 735-740.

Gaver, W. W., Smith, R. and O'Shea, T., 1991. "Effective sounds in complex systems: the ARKola simulation" In: CHI'91, New Orleans, Louisiana, USA, 1991. ACM Press, 85-90.

Golick, J., 1999. "Network computing in the new thin-client age" NetWorker, March, 31-40.

Gordan, A. D., 1981 Classification: Methods for the Exploratory Analysis of Multivariate Data, Chapman and Hall.

Gould, J., 1988, How to design usable systems, in: Handbook of Human-Computer Interaction, Ed, Helander, M., pp. 757-790, Amsterdam, Holland:North-Holland.

Grand, M., 1998 Patterns in Java - Volume 1, John Wiley & Sons, Inc., New York.

Hartigan, J. A., 1974 Clustering Algorithms, Wiley, New York.

Herman, I., Melancon, G. and Marshall, M. S., 2000. "Graph Visualization and Navigation in Information Visualization: a Survey" IEEE Transaction on Visualization and Computer Graphics, 6, 1, 24-43.

Hutchins, E., 1989, Metaphors for interface design, in: The Structure of Multimodal Dialogue, Eds, Taylor, M. M., Neel, F. and Bouwhuis, D., 11-28, Amsterdam:Elsevier Science Publishers.

Hutchins, E. L., Hollan, J. D. and Norman, D., 1986, Direct manipulation interfaces, in: User-Centred System Design, Eds, Norman, D. and draper, S., 87-124, Hillsdale, NJ, USA:Lawrence Erlbaum Associates.

Jog, N. and Shneiderman, B., 1994. "Starfield information visualization with interactive smooth zooming" In: IFIP 2.6 Visual Databases Systems, Lausanne, Switzerland, 1994.

Johnson, B. and Shneiderman, B., 1991. "Tree-maps: A space filling approach to the visualization of hierarchical information structures" In: IEEE Visualization '91, San Diego, CA, October 1991, 1991. 284-291.

Jose, J. M., Furner, J. and Harper , D. J., 1998. "Spatial querying for image retrieval: a user-oriented evaluation" In: ACM SIGIR conference on Research and development in information retrieval, Melbourne Australia, 1998. 232-240.

Kang, H. and Shneiderman, B., 2000. "Visualization Methods for Personal Photo Collections: Browsing and Searching in the PhotoFinder" In: IEEE International Conference on Multimedia and Expo (ICME2000), New York, 2000. IEEE, 1539-1542.

Kaufman, L. and Rouseeuw, P. J., 1990 Finding Groups in Data, Wiley, New York.

Knudsen, C., 1999. "MP3 Linux Players", Linux Journal, July,Article No. 10.

Kobayashi, M. and Schmandt, C., 1997. "Dynamic Soundscape: mapping time to space for audio browsing" In: CHI'97, Atlanta, GA, USA, 1997. ACM Press, 194-209.

Koffka, K., 1935 Principles of Gestalt Psychology, Harcourt-Brace, New York.

Kornstädt, A., 2001. "The JRing System for Computer-Assisted Musicological Analysis" In: International Symposium on Music Information Retrieval (ISMIR'01), Indiana, USA, October 15-17, 2001. 93-111.

Kramer, G. E., Walker, B., Bonebright, T., Cook, P., Flowers, J., Miner, N., Neuhoff, J., Bargar, R., Barrass, S., Berger, J., Evreinov, G., Fitch, W. T., Grohn, M., Handel, S., Kaper, H., Levkowitz, H., Lodha, S., Shinn-Cunningham, B., Simoni, M. and

Tipei, S., 1999, "The Sonification Report: Status of the Field and Research Agenda", ICAD-NSF Report Committee

Lam, C. K. M. and Tan, B. C. Y., 2001. "The Internet is changing the music industry" Communications of the ACM, 44, 8, 62-68.

Lamping, J., Rao, R. and Pirolli, P., 1995. "A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies" In: CHI '95, Denver, CO, USA, 1995. ACM Press, 401-408.

Leander, R., 2000 Building Application Servers, SIGS Books, New York, NY.

Leung, Y. K. and Apperley, M. D., 1994. "A Review and Taxonomy of Distortion-Oriented Presentation Techniques" ACM Transactions on Computer-Human Interaction, 1, 2, 126-160.

Lewis, C., 1982, "Using the "Thinking-aloud" Method in Cognitive Interface Design", RC 9265 IBM Yorktown Heights.

LoPresti, E. and Harris, W. M., 1996. "loudSPIRE, and Auditory Display Scema for the SPIRE System" In: ICAD '96, Palo Alto, CA, USA, 1996.

Macaulay, C. and Crerar, A., 1998. "'Observing' the Workplace Soundscape: Ethnography and Auditory Interface Design" In: ICAD '98, Glasgow, Scotland, 1-4/11/1998, 1998. Springer-Verlag,

Mackay, W., 2002, "Using Video to Support Interaction Design", 1-58113-516-5 ACM/SIGCHI

Marchionini, G., 1995 Information Seeking in Electronic Environments, The Press Syndicate of the University of Cambridge, New York, USA.

Marchionini, G. and Shneiderman, B., 1988. "Finding facts versus browsing knowledge in hypertext systems" IEEE Computer, 19, 70-80.

Masui, T., 1998. "LensBar - visualization for browsing and filtering large lists of data" In: IEEE Symposium on Information Visualization, 1998. 113 -120.

McCarthy, B. and Cassady-Dorion, L., 1998 Java Distributed Objects, SAMS, Indianapolis, Indiana, USA.

McLachlan, G. J. and Basford, K. E., 1988 Mixture Models: Inference and Applications to Clustering, Marcel Dekker.

Melih, K. and Gonzalez, R., 1998. "Audio retrieval using perceptually based structures" In: IEEE International Conference on Multimedia Computing and Systems, 1998. 338 -347.

Moffitt, J., 2001. "Ogg Vorbis—Open, Free Audio—Set Your Media Free", Linux Journal, January 2001,

Munzner, T., 1997. "H3: Laying Out Large Directed Graphs in 3D Hyperbolic Space" In: IEEE Symposium on Information Visualization, Phoenix, AZ, October 20-21, 1997. 2-10.

Munzner, T., 2000. Interactive Visualization of Large Graphs and Networks. Ph.D. Dissertation, Stanford University.

Murtagh, F., 1985 Multidimensional Clustering Algorithms, Physica-Verlag.

Mynatt, E. and Edwards, W., 1992, "The Mercator Environment: A Non-visual Interface to the X Window System", GIT-GVU-92-05 Georgia Institute of Technology Atlanta.

Mynatt, E. D., 1994. "Designing with Auditory Icons: How Well Do We Identify Auditory Cues ?" In: CHI'94, Boston, Massachusetts, USA, April 24-28, 1994. ACM Press, 269-270.

Nielsen, J., Clemmensen, T. and Yssing, C., 2002. "Getting access to what goes on in people's heads? - Reflections on the think-aloud technique" In: NordiCHI, Århus, Denmark, October 19-23, 2002. ACM, 119-128.

Ó Maidín, D., 1998. Introduction to C.P.N. View University of Limerick, Technical Report UL-CSIS-98-03.

Ó Maidín, D. and Fernström, J. M., 2000. "The Best of Two Worlds: Retrieving and Browsing" In: COST-G6 Conference on Digital Audio Effects DAFx-00, Verona, December 7-9, 2000. Rocchesso, D. and Signoretto, M.: Universita degli Studi Verona, 131-134.

Ó Maidín, D. S., 1995. A Programmer's Environment for Music Analysis. Ph.D., University College Cork.

Pauws, S., Bouwhuis, D. and Eggen, B., 2000. "Programming and Enjoying Music with Your Eyes Closed" In: CHI2000, The Hague, Amsterdam, April 1-6, 2000. ACM Press, 376-383.

Perrone, P. J. and Chaganti, V. R., 2000 Building Java Enterprise Systems with J2EE, SAMS, Indianapolis.

Perrot, D. and Gjerdigen, R. O., 1999. "Scanning the dial: An exploration of factors in the identification of musical style" In: Society for Music Perception and Cognition, 1999. 88(abstract).

Pook, S., Lecolinet, E., Vaysseix, G. and Barillot, E., 2000a. "Context and Interaction in Zoomable User Interfaces" In: Advanced Visual Interfaces (AVI 2000), Palermo, Italy, 2000a. ACM Press, 227-231.

Pook, S., Lecolinet, E., Vaysseix, G. and Barillot, E., 2000b. "Control Menus: execution and control in a single interactor" In: Computer Human Interaction (CHI) '00, The Hague, The Netherlands, Apr 2000, 2000b. ACM Press,

Pook, S., Vaysseix, G. and Barillot, E., 1998. "Zomit: biological data visualization and browsing" Bioinformatics, 14**,** 9**,** 807-814.

Preece, J., Rogers, Y. and Sharp, H., 2002 Interaction Design: Beyond Human-Computer Interaction, John Wiley & Sons.

Prothman, B., 2000. "Meta data" IEEE Potentials, 19**,** 1**,** 20 -23.

Ralston, A., Reilly, E. D. and Hemmendinger, D. (Eds.) 2000. Encyclopedia of Computer Science,Grove's Dictionaries, Inc., New York, NY.

Raskin, J., 2000a The Human Interface: New Directions for Designing Interactive Systems, Addison Wesley.

Raskin, J., 2000b The Humane Interface, ACM Press / Addison - Wesley, Harlow, England.

Rath, M., 2002, "Models and Algorithms for Sounding Objects: Sound design around a real-time impact model", Deliverable n.6

Richter, J., 2000. "Microsoft .NET Framework Delivers the Platform for an Integrated, Service-Oriented Web", MSDN Magazine, September,

Risen, K., Czerwinski, M. P., Munzner, T. and Cook, D., 2000. "An Initial Examination of Ease of Use for 2D and 3D Information Visualizations of Web Content" International Journal of Human Computer Studies.

Robertson, G. G., Mackinlay, J. D. and Card, S. K., 1991. "Cone Trees: Animated 3-D Visualizations of Heirarchical Information" In: CHI'91, New Orleans, Louisiana, USA, 1991. ACM Press, 189.

Rodden, K., Basalaj, W., Sinclair, D. and Wood, K., 2001. "Does organisation by similarity assist image browsing?" In: SIG-CHI on Human factors in computing systems, Seattle, WA USA, March 31 - April 5, 2001. 190-197.

Sarkar, M. and Brown, M. H., 1992. "Graphical fisheye views of graphs" In: Human Factors and Computing Systems, Monterey, California, United States, 1992. ACM Press, 83-91.

Scaletti, C., 1994, Sound Synthesis: Algorithms for Auditory Data Representation, in: Auditory Display: Sonification, Audification and Auditory interfaces, Ed, Kramer, G., 223-252, Reading, MA, USA:Addison-Wesley Publishing Company.

Serafine, M. L., 1988 Music as cognition : the development of thought in sound, Columbia University Press, New York, USA.

Shneiderman, B., 1992a Designing the User Interface: Strategies for Effective Human-Computer Interaction, Addison-Wesley, Reading, MA, USA.

Shneiderman, B., 1992b. "Tree visualization with tree-maps: A 2-d space-filling approach" ACM Transactions on Computer Graphics, 11, 1, 92-99.

Sonnenschein, D., 2001 Sound Design: The Expressive Power of Music, Voice, and Sound Effects in Cinema, Michael Wiese Productions, Studio City, CA, USA.

Sprenger, T. C., Brunella, R. and Gross, M. H., 2000. "H-BLOB: A Hierarchical Visual Clustering Method Using Implicit Surfaces" In: IEEE Visualization '00, 2000.

Sprenger, T. C., Gross, M., Bielser, D. and Strasser, T., 1998. "IVORY: An Object Oriented Framework for Physics Based Information Visualization in Java" In: IEEE Symposium on Information Visualization (InfoViz'98), 1998. IEEE CS Press,

Sun Microsystems, I., 2002. Java Remote Method Invocation - Distributed Computing for Java Website.

Tzanetakis, G., 2002. Manipulation, Analysis And Retrieval Systems For Audio Signals. PhD Disseration, Princeton University.

Tzanetakis, G. and Cook, P., 2000a. "3D graphics tools for sound collections" In: COST-G6 Conference on Digital Audio Effects DAFx-00, Verona, December 7-9, 2000a. Rocchesso, D. and Signoretto, M.: Universita degli Studi Verona, 115-118.

Tzanetakis, G. and Cook, P., 2000b. "Audio Information Retrieval (AIR) Tools" In: International Symposium on Music Information Retrieval (MUSIC IR 2000), Plymouth, Massachusetts, 2000b. University of Massachusetts,

Tzanetakis, G. and Cook, P., 2000c. "MARSYAS: a framework for audio analysis" Organised Sound, 4, 3.

Tzanetakis, G. and Cook, P., 2001. "MARSYAS3D: a Prototype Audio browser-Editor using a Large Scale Immersive Visual and Audio Display" In: ICAD 2002, Helsinki, Finland, July 29 - August 1, 2001. Hiipakka, J., Zacharov, N. and Takala, T.: Laboratory of Acoustics and Audio Signal Processing and the Telecommunications Software and Multimedia Laboratory, Univeristy of Helsinki, Espoo, Finland, 250-254.

Waldo, J., 1998. "Remote procedure calls and Java Remote Method Invocation" IEEE Concurrency, 6, 3, 5-7.

Walker, A., Brewster, S. A., McGookin, D. and Ng, A., 2001. "Diary in the sky: A spatial audio display for a mobile calendar" In: BCS IHM-HCI 2001, Lille, France, 2001. Springer, 531-540.

Walker, B., 2000. Magnitude Estimation of Conceptual Data Dimensions for Use in Sonification. Ph.D. Dissertation, Rice University.

Warren, R. M., 1999 Auditory Perception - A new Analysis and Synthesis, Cambridge University Press, Cambridge.

Welsh, M., Borisov, N., Hill, J., von Behren, R. and Woo, A., 1999, "Querying large collections of music for similarity", UCB/CSD00-1096 Computer Science Division, U.C Berkeley

Wenzel, E. M., 1994, Spatial Sound and Sonification, in: Auditory Display: Sonification, Audification and Auditory interfaces, Ed, Kramer, G., 127-150, Reading, MA, USA:Addison-Wesley Publishing Company.

Wickens, C. D. and Hollands, J. G., 2000 Engineering Psychology and Human Performance, Prentice Hall, New Jersey, USA.

Williamson, C. and Shneiderman, B., 1992. "The Dynamic Homefinder: evaluating dynamic queries in a real estate information exploration system" In: Special Interest Group on Information Retrieval SIGIR '92, 1992. ACM Press, 339-346.

Wold, E., Blum, T., Keislar, D. and Wheaton, J., 1996. "Content-based classification, search and retrieval of audio" IEEE Multimedia, 3, 2, 27-36.

Woods, D. D., Patterson, E. S., Roth, E. M. and Christoffersen, K., 1999. "Can we ever escape from data overload?" In: 43rd Annual Meeting on Human Factors and Ergonomics Society, Houston, Texas, 1999.

Yewdall, D. L., 1999, Sound Librarian: Curator of an Audio Empire, in: Practical Art of Motion Picture SoundButterworth-Heinemann.

Zanella, A., Carpendale, M. S. T. and Rounding, M., 2002. "On the Effects of Viewing Cues in Comprehending Distortions" In: NordiCHI, Århus, Denmark, October 19-23, 2002. ACM, 119-128.

Zhang, T., Kuo, C.C., 1998a. "Content-based Classification and Retrieval of Audio" In: SPIE's 43rd Annual Meeting - Conference on Advanced Signal Processing Algorithms, Architectures, and Implementations VIII, San Diego, July, 1998a. 432-443.

Zhang, T., Kuo, C.C., 1998b. "Hierarchical System for Content-Based Audio Classification and Retrieval" In: SPIE's Conference on Multimedia Storage and Archiving Systems III, Boston, 1998b. p398-409.

Appendix A: Technology Review

In this section, we present a brief overview of the network mechanisms suitable to the creation of a client server application. From networking research there are various mechanisms for the moving a standalone application to a distributed application. The various models boil down in essence to a couple of core concepts, the client-server architecture and the mobile code architecture. The client-server architecture is a more established research area than the mobile code architecture. The difference between the two is that the client-server focuses on two processes: a client and a server that run separately whereas the mobile code is a complete application process that migrates across the network. Most distributed architectural styles either use or are in part based upon Remote Procedure Call (RPC) for communications among distributed components (Perrone and Chaganti, 2000). Middleware infrastructures such as CORBA and Java + Remote Method Invocation (RMI) (Sun Microsystems, 2002) are based on this kind of communication model (Perrone and Chaganti, 2000). RPC is based on a tight coupling between the object that requests a service (i.e., the client) and the object that satisfies such request (i.e., the server). Before invoking a service, the client has to know the existence of a server capable of satisfying its request and has to obtain a reference to such server. These areas will be discussed in greater detail in the following sections. Clients vary depending upon processing ability, which determines their suitability for being either 'thin' or 'fat'. The 'thin' clients don't process information but only present the information while 'fat' clients can further processes information and also display it. The 'thin' client is the architecture for the client side of the present application.

Several related applications using many of the technologies described in are studied in much greater detail. The *Sonic Browser* system is covered in section 3.7, which is a tool for browsing sounds, or collections of sounds, using sound spatialization and context-overview visualisation techniques. It is this system which we have based our prototype system upon. A view of how our prototype system will utilise the client-server architecture is illustrated in Figure 35 with the server/s on the right of the diagram and the client/s on the left, separated by a network layer. The client aspect of our prototype system will handle the presentation layer, which consists of the user interface (UI) and

the sound processing elements. The server aspect of our prototype system handles the database and communication layers providing persistent storage of sound and message information.

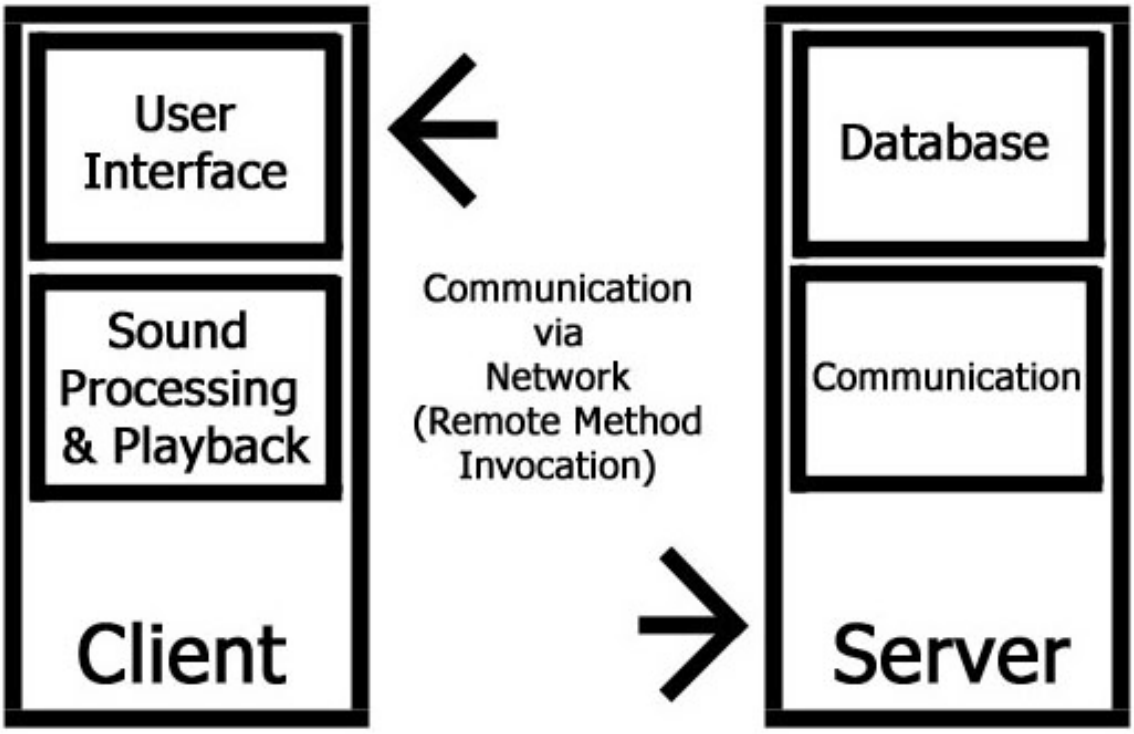


Figure 35: Overview of Client-Server Architecture as used within the Sonic Browser

What is a thin Client - Server Application?

The Client-Server architecture has several definitions. Furht el at (2000) p68 defines *client-server computing* as "clients are powerful and processing is centered around local execution on clients. Computer resources were split between a server and one or several clients". Golick (1999) p34 defines a thin-client system as "a cooperative processing environment that primarily implements applications based on remote and distributed presentation designs", which is illustrated in Figure 36. A simpler definition of it is as an architecture where two or more computers or processes interact in such a way that one provides services to the other. Ralston et al (2000) p1 states that the "Client/server is a distributed computing model in which *client* applications request services from *server* processes. Clients and servers typically run on different computers interconnected by a computer network. A *client* application is a process or program that sends messages to a

server via the network. The *server* process or program listens for client requests that are transmitted via the network."

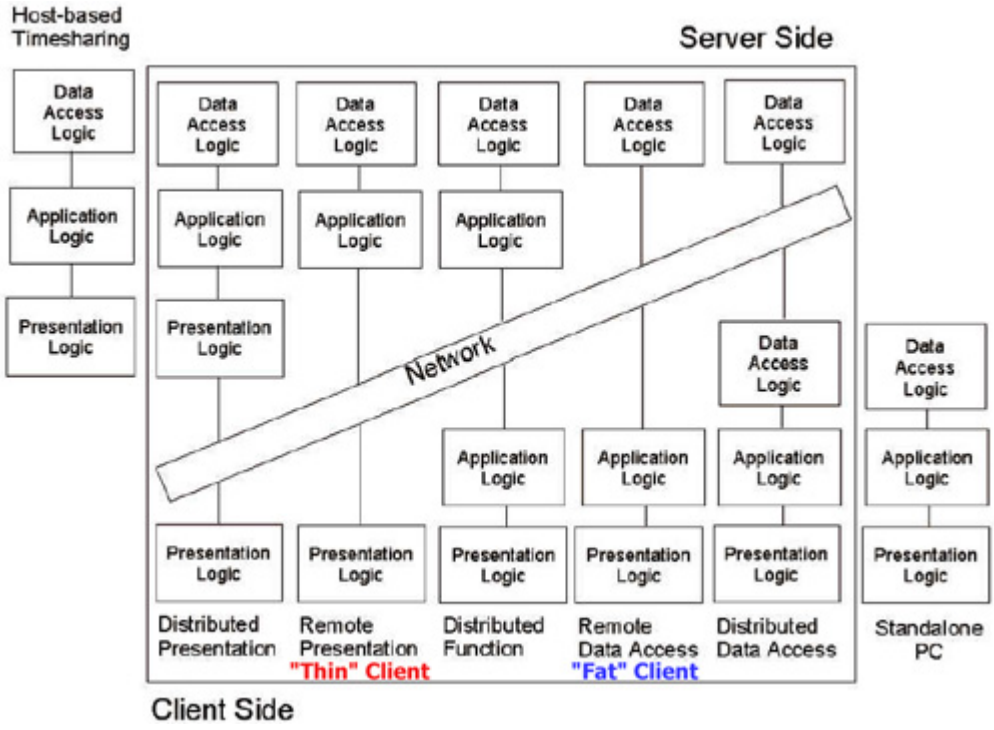


Figure 36: Possible Client-Server Configurations (Source: Gartner Group)

This research focuses on the definition by Ralston as the more broadly accepted definition for the client-server architecture. The term client-server has also been applied to peer-to-peer computing at times. This, while technical correct since a peer-to-peer contains both a server and a client element's, is more correctly referred to as a distributed architecture by McCarthy et al (1998) p316 as "one that includes multiple servers" but the difference with client-server is that to "be considered truly distributed, a system should include multiple concurrent server connections". An example of a simple distributed architecture is peer-to-peer networking where it is possible for every host to act as both client and server.

Remote Method Invocation

Remote Method Invocation (RMI) (Sun Microsystems, 2002) (Waldo, 1998).is a mechanism within the Java programming environment that allows objects in different

Java virtual machines on the same or different hosts to send and receive messages. This is achieved by making the remote object's appear as local objects on a host.

The RMI model in Java allows a client machine to invoke a method on a remote server machine in similar, but not quite identical manner to normal method invocation. A remote object on the server is the class instance of specific class, which is presented using the special Remote interface. The server firstly registers its remote interface with a known registry; the client then looks this registry to locate the desired object. During the compilation phase on the client, a set of stub functions are created, when the client calls the function, these stubs forward the call to the server to be processed using the methods on the server.

Jini

The Jini environment uses Java's ability to move objects including each object's data and code from one Java environment to another. The Java environment features include portable bytecodes; dynamic loading of code, code verification, and security "sand boxing" allows such an approach straightforwardly. Adding to the existing Java environment the Jini programming model provides a toolkit containing several basic building blocks required for distributed applications: distributed events, transactions, leases, and downloadable proxies. The mechanisms for using these building blocks in Jini include a set of conventions for advertising and finding services on the network, a service that acts as a place to do such service advertising and finding, and a means for bootstrapping the system without human intervention. In a Jini network, services advertise themselves by saying what Java language interfaces they implement. Clients look for services by specifying what Java language interfaces they need.

This matching of client and service differs from more usual naming or directory lookups in that it allows the services to evolve over time without losing the ability to be used by clients that only expect the functionality that the service previously provided. Advertisement and matching of client and service occurs in the a lookup service which are found using the Jini discovery protocol, a place where providers of a service can register what they provide and clients of services can look for what they need. The service actually stores the information needed to reconstruct a Java object as a proxy for

the service in the client's virtual machine. This approach means that all the client need know about the service is the interface the service provides; everything else (such as how the proxy and the service communicate) is hidden behind the implementation of the proxy object.

Middleware Client/Server Communication

Bacon et al (2000) p68 define middleware as " a layer of software that runs above heterogeneous operating systems and communications systems, providing a uniform interface to distributed applications". Abdul-Fatah et al (1998) p1 define middleware as "a software layer that resides between the application and the operating system, has been introduced to facilitate the design and implementation of client-server systems in a heterogeneous environment". Leader (2000) p18 defines middleware as "a category of software that provides program-to-program communication across multiple computers". A number of different types of models have been proposed to standardize the communication middleware.

JAVA

Java offers a wide selection of network API's and also offers a number of graphical user interface (GUI) libraries that make it a possible candidate for the creation of our application. Java's core foundation also offers several language, and library extensions, useful for application development shown in Table 27. An overview diagram of the Java language features for version 1.4 is shown below in Figure 37. The Java language is an object-oriented language, which uses a syntax similar to C++. A notable difference with C++ is Java's automatic garbage collection whereby dynamic allocated memory is freed once it has been detected that it is no longer being used. Platform independence is achieved by the intermediate "bytecode" representation, which can be exchanged and executed by Java Virtual Machines (JVMs) on almost any computing platform. A major performance issue with the first JVMs were that simply interpreting the bytecode of a Java program which resulted in very poor performance for Java programs. There have been extensive developments in the past few years in this area and today nearly every JVM uses just-in-time (JIT) compiler technology. JITs operate as part of the JVM,

compiling Java byte- code into native machine code at runtime. Once the JVM generates the machine code, it executes it at raw machine code speed. Modern JITs feature many sophisticated optimisations techniques and methods such as array bounds check elimination, method devirtualisation and stack allocation of objects. The Hotspot™ (Boisvert et al., 2001) compiler and runtime aspects are Sun's latest version of this JIT compiler technology and can be seen within the platform structure of Java 1.4 towards the bottom of Figure 37.

Swing	Graphical components but "all Java" or platform independent.
AWT	Graphical components used for creating user interfaces and for painting graphics and images but may be platform dependent for the actual implementation.
Java 2D & 3D	Graphical drawing components used for both 2D and 3D drawing primitives.
Util	Miscellaneous container components such as Lists, Arrays and Maps, date and time facilities, internationalisation, and miscellaneous utility classes (a string tokeniser, a random-number generator, and a bit array).
Net	Provides the classes for implementing networking applications.
Sql	Provides the classes for accessing and processing data stored in a data source (usually a relational database) using the Java™ programming language.

Table 27: Useful Java Library Extensions

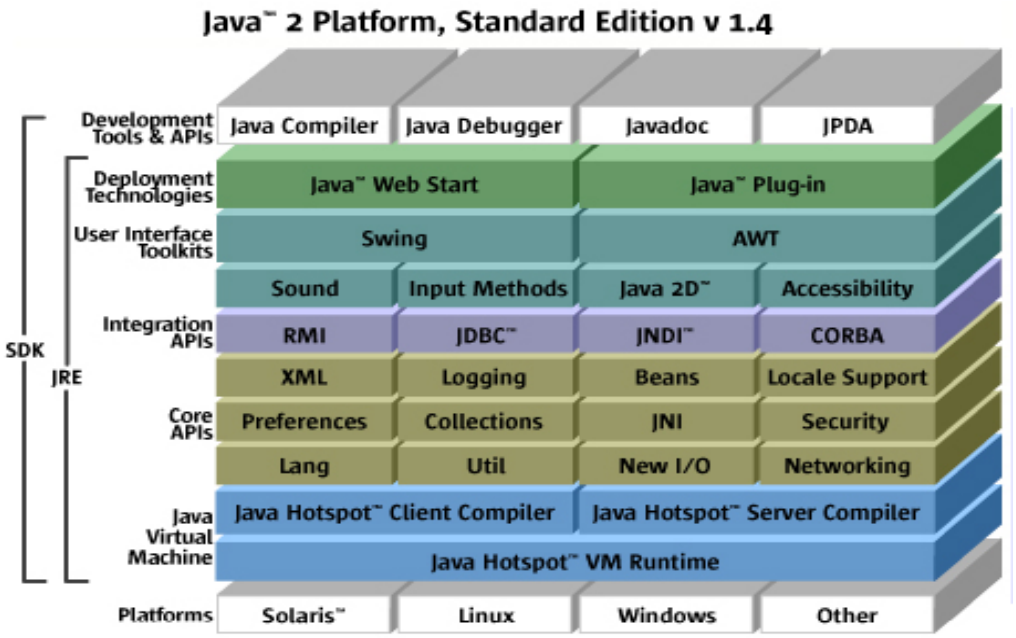


Figure 37: The Java Platform version 1.4 (Source: Sun Microsystems Inc.)

Appendix B: Architectural Styles & Patterns

In this section, a brief description of the important patterns used within our prototype system is provided; this is followed by a more detailed explanation of these patterns with their specific use within the context of our application. The three patterns that are focused on are the Singleton pattern, the Interface pattern and the Mediator pattern. These have been chosen due to their use in the three major sections of the application (Database, Graphics and Communications). Understanding this rationale should provide a greater degree of illumination and understanding of the specific use in our prototype system.

The Singleton pattern (Gamma et al., 1995) is a pattern that ensures that only one instance of a class is created. In an application, all objects created that use an instance of that class will use the same instance. This pattern is used for keeping sound resource information and passing variables amongst several objects but still ensuring that they all receive the same information as they all use the same object instantiation of the class. The Singleton pattern is discussed in Appendix D.

The Interface pattern (Grand, 1998) is a pattern that allows the separation of a class's functionality from its implementation. This ensures that classes, which use the data and services of the Interface's class instances, are kept independent of its particular implementations. The classes accessing it are unaware of a particular implementation of the class's functionality so any changes to the implementation are invisible to the classes accessing it, these are the class instances which access it via an interface class. This pattern is used by the communication and the database elements of our prototype system that allows for the creation of different classes underlying these interfaces, which implement these interfaces. The different classes can be new classes, which have been developed to utilise a new database system or a new type of communications subsystem. This allows for the future updating of these sections without affecting the rest of the application. The Interface pattern is discussed in Appendix E.

The Mediator pattern (Gamma et al., 1995) is a pattern that uses one object to coordinate state changes between objects. This pattern localises all the logic in one object that manages the state changes of other objects, resulting in a more cohesive implementation of the logic and a reduced coupling between the objects. The Mediator pattern is discussed in Appendix F.

Appendix C: Implementation and related problems

In this section, a brief description of the important implementation issues within the application is provided; this is followed by an explanation of the problems with their specific implementation within the context of our prototype system. The three areas of implementation that are focused on are the Database subsystem, the Graphical User Interface subsystem and the Communication subsystem.

Database Subsystem Implementation

The main processor class for the database functionality is the SobDbHandler class. It has all the database functionality for our prototype system including persistent storage for all user changes as well as initial properties of sounds such as perceptual, technical and user defined classifications. It also provides functionality for the querying for particular sounds based on arbitrary properties.

Database Implementation Problems

Adding persistence was a major change to the design of previous versions of our prototype system. A major issue in the database area was choosing the necessary information and database structure which would be required to store this information. The final database structure can be seen in Appendix D. The greatest difficulty that was encountered in this area was the mapping of objects to statements that would then be performed on the database. At the beginning of the project several database to object mapping solutions were tested but they did not lend themselves to easy implementation or use. This lead to the implementation of a custom application to database mapping logic which should be avoided if at all possible by use of a database to object mapping software if suitable and available.

Graphical User Interface Subsystems

The main class for the application is the SobFrame but this class is really just a frame for holding the graphical user interface primary class, SobOverviewLayout. This class holds the relevant controls and logic for the layout and appearance of our prototype system. This is used to display and edit details about a particular node. These classes use the information in the SobFileModel, which holds a collection of SobFileNode. Each of the SobFileNodes contains all the information about each particular sound. All the visualisations exploit the Model-View-Controller design pattern (Friedrichs et al., 1999) to allow multiple views of the same underlying data to be accessed and modified from the SobFileModel. It is this model of the data which is critical for the consistency in presentation across the various visualisations while still allowing for the use of dynamic queries, which also maintain this consistency across the visualisations.

Graphical User Interface Problems

The graphical user interface offered many problems due to the lack of a what-you-see-is-what-you-get (WYSIWYG) editor for the interface components. This is due to the fact that the application was built using an open-source solution that did not offer this functionality. The problems encountered were mainly related to inconsistency across layouts and while not difficult to solve, it was time-consuming to achieve solutions that produced functional and aesthetic layouts.

Communication Subsystem

The system used for the communications layer within our prototype system uses several objects. A SobUserID object represents the user; this is used for identification and ownership purposes. Each user accesses a SobCollaborator object for communication, which acts as the client side for communications. The SobCollaborator object communicates to the server that is represented by a SobMediator object. Each server object is represented a unique identifier object, a SobMediatorID object. An additional layer has been added to the communication layer to form a client – server – super server hierarchy. A SobSuperMediator object represents the super server object.

There are five steps in the communication process for our prototype system. The first step is for the SobCollaborator to register for a SobUserID. The second step is for the next valid SobUserID number to be issued to the SobCollaborator which requested registration. The third step is for the SobCollaborator to register with the server or SobMediator with an identifier, which contains the particular SobCollaborator's SobUserID number. The fourth step is the most common step for the SobCollaborator, where is passes any changes or messages to the SobMediator. The fifth step is where the SobMediator receives the message, processes it and depending on the message it may forward it to all the collaborators for use. The message will be received by the SobCollaborator, which sent it, but in this case it is simply discarded. The sequence of events in the communication process is shown in Figure 38. A similar but untested portion of functionality exists for a super server architecture, which replaces the Collaborator with a Mediator, and a SuperMediator takes the vacant Mediator position. This is a future mechanism for processing load balancing across servers.

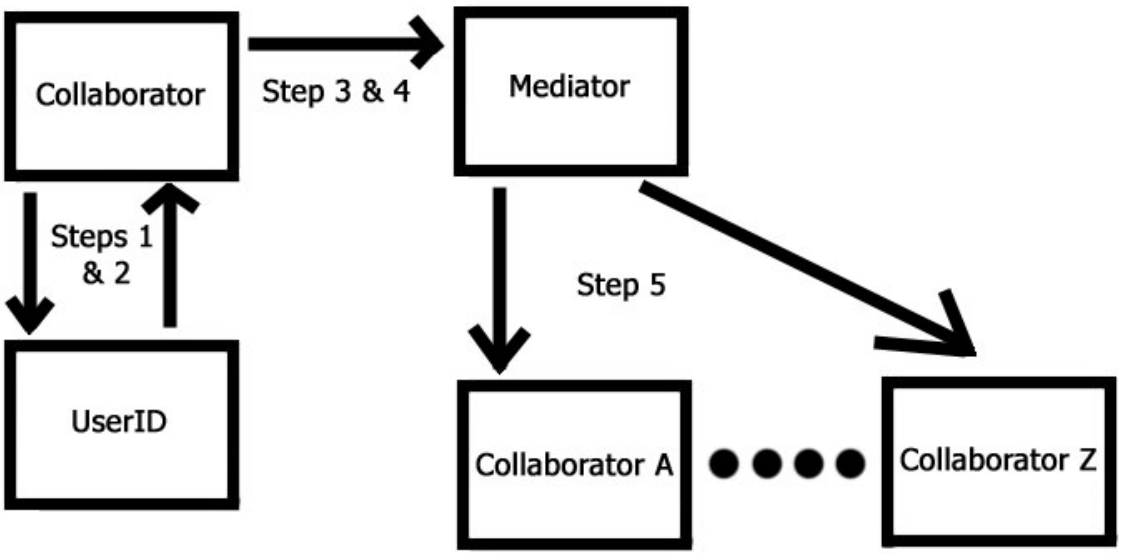


Figure 38: Operation of Communication Subsystem

Communication Problems

The communication system offered less implementation problems than other areas but these were surpassed by the difficulties in testing distributed software. This type of testing requires the use of several networked computers, which should be factored into the timetable and resource requirements for testing.

Appendix D: Singleton Pattern

Amongst several classes that use this pattern are SobGeneralConstants, SobShapeConstants and SobResourceConfigConstants for holding globally accessible variables and lookup tables for particular values. These classes are generally configuration classes and allow for a static global object instance. SobWaveSoundDecoder, SobAiffSoundDecoder and SobAuSoundDecoder are special sound file decoder classes which take a file and parse it for important header information such as details on cue points as well as sampling rate, the number of channels and other specific technical details regarding the sound file. These are the two different categories that the Singleton pattern has been used for within our prototype system. Firstly, for global variables or lookup tables and then for specific functions, both are in essence standalone global black box functions.

An abbreviated version of the SobGeneralConstants class is shown on the next page with the relevant sections for creating a single object instance in Java.

```
public final class SObGeneralConstants
{
   private static SObGeneralConstants GeneralConstants = null;

   ……………
   private SObGeneralConstants()
   {

   ……………
   }

   ……………
   /**
   * This method returns an instance of the SObGeneralConstants.
   * If there is no instance then it will create a new one.
   * In order to use the SObGeneralConstants do the following:
   * private SObGeneralConstants GeneralConstants = SObGeneralConstants();
   *
   * @return an instance of the SObGeneralConstants
   */
   public synchronized static SObGeneralConstants getInstance()
   {
      if (GeneralConstants != null)
      {
         return GeneralConstants;
      }

      GeneralConstants = new SObGeneralConstants();

      return GeneralConstants;
   }

   ……………
}
```

Appendix E: Interface Pattern

The Interface pattern is prominently found in the classes of SobCollaborator, SobMediator, SobSuperMediator and SobDbInterface. These are split into two grouping with the first three classes belonging to the networking subsystem while the SobDbInterface belongs to the database subsystem. The Interface code is used in the context of allowing the underlying functionality of the objects to be later modified to take advantages of technological or algorithmitic improvements. In the case of the networking subsystem, a replacement communication protocol or technology could be implemented without adversely affecting the entire application and similarly so for the database subsystem.

An abbreviated version of the SobCollaborator class is shown on the next page with the relevant sections for creating a single object instance in Java.

```
public interface SObCollaborator
   extends Remote
{
   public SObUser getSObUser()
                  throws RemoteException;

   // Connect to a local mediator - subclasses dictate properties needed
   public boolean connect()
                throws RemoteException;

   // Queries connection status
   public boolean isCommConnected() throws RemoteException;

   // Outgoing messages/data
   public boolean send(String tag, String msg, SObUser dst)
              throws IOException, RemoteException;

   public boolean send(String tag, Object data, SObUser dst)
              throws IOException, RemoteException;

   public boolean broadcast(String tag, String msg)
                throws IOException, RemoteException;

   public boolean broadcast(String tag, Object data)
                throws IOException, RemoteException;

   // Incoming messages/data
   public boolean notify(String tag, String msg, Date mtime, SObUser src)
              throws IOException, RemoteException;

   public boolean notify(String tag, Object data, Date mtime, SObUser src)
              throws IOException, RemoteException;
}
```

Appendix F: Mediator Pattern

The Mediator pattern is used within the networking subsystem in the server side classes to mediate state changes between the user objects and the server. This allows for future updating of the logic within the networking subsystem without producing errors, which occur at the scope of the application since the logic has been centralised to the server classes within the networking subsystem.

The broadcast function from the SobMediator class is shown on the next page with the relevant sections for passing a string message to all the other clients registered with the server.

```
   /**
    * broadcast() takes parameters for a message and sends it to all the other users
registered to the mediator
    * @param SObUser from Date mtime String mtag String msg
    * @return boolean
    */
   public boolean broadcast(SObUser from, Date mtime, String mtag, String msg)
                throws IOException, RemoteException
   {


      boolean success = true;
      Enumeration ids;
      Enumeration collaborators;
      System.out.println(
            "Preparing to BROADCAST message from " + from.getUserID());

      /* Get the list of members registered to the mediator and using cloning to reduce
demand */
      Hashtable c_members = (Hashtable)members.clone(); //Clone to reduce
synchronisation requirements
      ids = c_members.keys();
      collaborators = c_members.elements();

      if (!ids.hasMoreElements())
      {

         // ("No clients to notify of new broadcast");
         return success;
      }

      /* Set up to spool the list of collaborator registered and then send the message to
each on the list */
      SObCollaborator target = null;

      while (ids.hasMoreElements())
      {

         /* The current collaborator on the list */
         SObUser i = (SObUser)ids.nextElement();
         target = (SObCollaborator)collaborators.nextElement();

         try
         {

            //target = (SObCollaborator)c_members.get(i);
```

```
            target.notify(mtag, msg, mtime, from);
            System.out.println(
                  "Sending message: \"" + mtag + " " + msg + "\"" +
                  " from " + from.getUserID() + " to " +
                  i.getUserID() + " sent at " + mtime);

            //reset timeout so do not penalize next client for previous
            //time client's
         }
         catch (RemoteException ex)
         {

            //not valid anymore so remove it from client list
            remove(i);
         }
         catch (Exception ex) {}
      }

      return success;
   }
```

Appendix G: Database Structure

```
-- MySQL dump 8.21
--
-- Host: localhost    Database: SObSounds
---------------------------------------------------------
-- Server version       3.23.49-nt

--
-- Table structure for table 'action'
--

CREATE TABLE action (
  ResourceID bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Description varchar(254) NOT NULL default '',
  PRIMARY KEY  (ResourceID,UserID,Description)
) TYPE=MyISAM;

--
-- Table structure for table 'axis'
--

CREATE TABLE axis (
  AxisProperty varchar(20) NOT NULL default '',
  Axis char(1) NOT NULL default '',
  UserID bigint(20) NOT NULL default '0',
  PRIMARY KEY  (Axis,UserID)
) TYPE=MyISAM;

--
```

```
-- Table structure for table 'environmental'
--

CREATE TABLE environmental (
  ResourceID bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Description varchar(254) NOT NULL default '',
  PRIMARY KEY  (ResourceID,UserID,Description)
) TYPE=MyISAM;

--
-- Table structure for table 'filters'
--

CREATE TABLE filters (
  FilterCategory varchar(20) NOT NULL default '',
  FilterProperty varchar(254) NOT NULL default '',
  UserID bigint(20) default NULL,
  PRIMARY KEY  (FilterCategory,FilterProperty)
) TYPE=MyISAM;

--
-- Table structure for table 'gridsize'
--

CREATE TABLE gridsize (
  TimeAtSize bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Height int(11) NOT NULL default '0',
  Width int(11) NOT NULL default '0',
  PRIMARY KEY  (TimeAtSize,UserID)
```

```
) TYPE=MyISAM;

--
-- Table structure for table 'music'
--

CREATE TABLE music (
  ResourceID bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Description varchar(254) NOT NULL default '',
  PRIMARY KEY  (ResourceID,UserID,Description)
) TYPE=MyISAM;

--
-- Table structure for table 'onomatopoeia'
--

CREATE TABLE onomatopoeia (
  ResourceID bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Description varchar(254) NOT NULL default '',
  PRIMARY KEY  (ResourceID,UserID,Description)
) TYPE=MyISAM;

--
-- Table structure for table 'owners'
--

CREATE TABLE owners (
  ResourceID bigint(20) NOT NULL default '0',
  private char(1) default NULL,
```

```
  created_time datetime NOT NULL default '0000-00-00 00:00:00',
  UserID bigint(20) NOT NULL default '0',
  password varchar(50) default NULL,
  PRIMARY KEY  (ResourceID,UserID)
) TYPE=MyISAM;

--
-- Table structure for table 'profiles'
--

CREATE TABLE profiles (
  ResourceID bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Name varchar(254) NOT NULL default '',
  InitalLayout char(1) NOT NULL default '',
  Time bigint(20) NOT NULL default '0',
  Shape varchar(254) NOT NULL default '',
  Color int(11) NOT NULL default '0',
  Xposition decimal(17,15) NOT NULL default '0.000000000000000',
  Yposition decimal(17,15) NOT NULL default '0.000000000000000',
  Zposition decimal(17,15) NOT NULL default '0.000000000000000',
  Tagged tinyint(1) NOT NULL default '0',
  PRIMARY KEY  (ResourceID,UserID,Time)
) TYPE=MyISAM;

--
-- Table structure for table 'resources'
--

CREATE TABLE resources (
  ResourceID bigint(20) NOT NULL default '0',
```

```
  UserID bigint(20) NOT NULL default '0',
  Name varchar(254) NOT NULL default '',
  SoundResourcePath varchar(254) NOT NULL default '',
  Description varchar(254) default NULL,
  FileType varchar(10) NOT NULL default '',
  PRIMARY KEY  (ResourceID,UserID,Name)
) TYPE=MyISAM;

--
-- Table structure for table 'source'
--

CREATE TABLE source (
  ResourceID bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Description varchar(254) NOT NULL default '',
  PRIMARY KEY  (ResourceID,UserID,Description)
) TYPE=MyISAM;

--
-- Table structure for table 'speech'
--

CREATE TABLE speech (
  ResourceID bigint(20) NOT NULL default '0',
  UserID bigint(20) NOT NULL default '0',
  Description varchar(254) NOT NULL default '',
  PRIMARY KEY  (ResourceID,UserID,Description)
) TYPE=MyISAM;
```

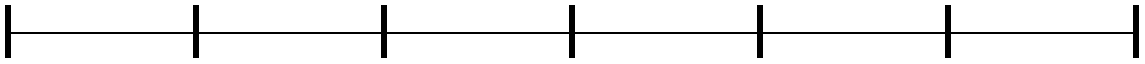
Appendix H: Experiment's Questionnaires

Sound Library Scenario

Each of the users was given a set of tasks to examine for a few minutes and then asked to complete those tasks using our prototype system. After the application was evaluated and the tasks completed by the user, a series of sixteen questions were asked as a post evaluation questionnaire. These were adapted from a "Subjective Usability Scales for Software - Sample Form" given in p550, "Software for Use". The user group consisted of approximately 2 users for the pilot study and 6 users for the evaluation study.

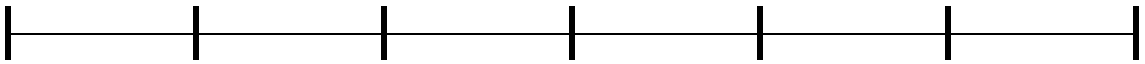
Q1: Understand this system?

hard - 0 ***to*** ***easy – 6***



Q2: Learn to use this system?

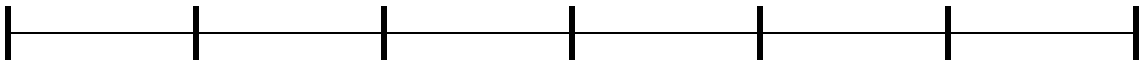
hard - 0 ***to*** ***easy – 6***



Based on your understanding of the design above, rate how easy you think it would be to use this application to accomplish each of the following tasks:

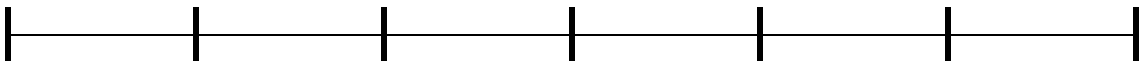
Q3: To find a particular sound, when you *know* its filename?

impossible - 0 ***to*** ***easy - 6***



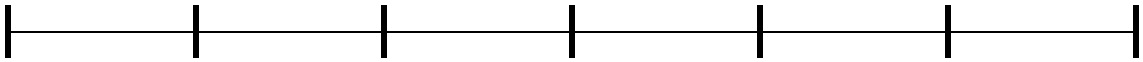
Q4: To find a particular sound, when you *don't know* its filename?

impossible - 0 ***to*** ***easy - 6***



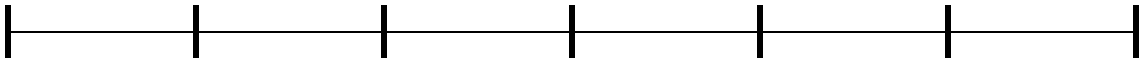
Q5: To find a set of particular sounds with a specific property from a category? (e.g.: Male voices from Speech category or Jazz from Music category)

impossible - 0 ***to*** ***easy - 6***



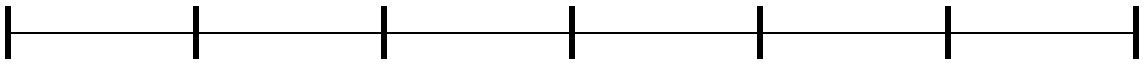
Q6: To find a range of sounds with any of a series of properties?

impossible - 0 ***to*** ***easy - 6***



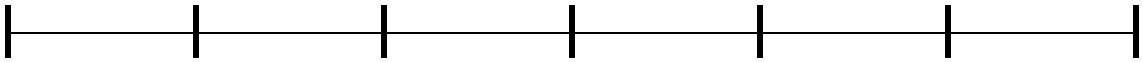
Q7: To find a particular sound when tagged?

impossible - 0 ***to*** ***easy - 6***



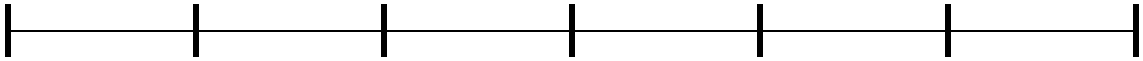
Q8: To find a particular sound, when you knew its shape?

impossible - 0 ***to*** ***easy - 6***



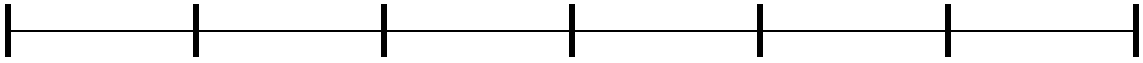
Q9: To perform an AND or an OR query using the filtering mechanism?

impossible - 0 ***to*** ***easy - 6***



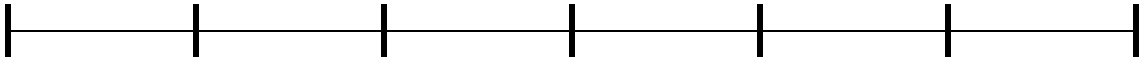
Q10: To change the details (Shape, Colour, Properties) of a particular sound?

impossible - 0 ***to*** ***easy - 6***



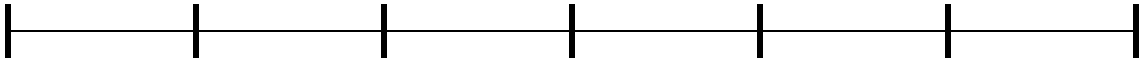
Q11: To select a group of sounds?

impossible - 0 ***to*** ***easy - 6***



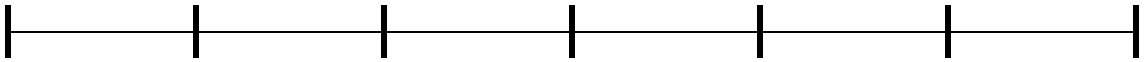
Q12: To play the last sound that you listened to?

impossible - 0 ***to*** ***easy - 6***



Q13: Overall, how would you rate the ease of use of this system?

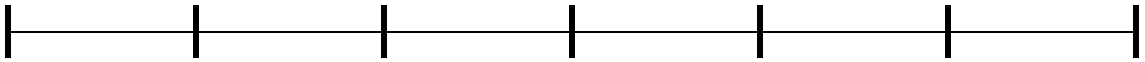
difficult - 0 ***to*** ***easy - 6***



Based on your perceptions of the sounds used, answer the following questions regarding the sounds:

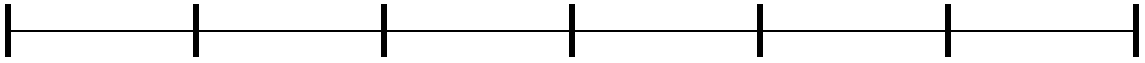
Q14: How realistic did you find the sounds?

artificial - 0 ***to*** ***realistic - 6***



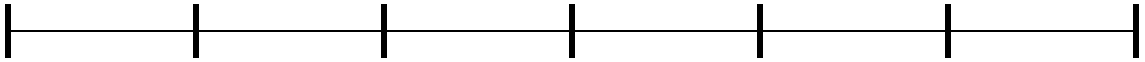
Q15: How did you find the quality of the sounds?

poor - 0 ***to*** ***superior - 6***



Q16: Was there a noticeable play lag before a sound was played?

obvious delay- 0 ***to*** ***no delay - 6***



Reference:

Software for Use - A practical guide to the models and methods of usage centered design**, 1999, ACM Press / Addison - Wesley, Larry L. Constantine and Lucy A.D. Lockwood.**

Appendix I: Sample User Report File

START SOb Sonic Browser - User Report File - 11-12-2002 05:06:29

ID,Color,Shape,X,Y,Sound Type,Filename

1,-1,1,0.276025236593059,0.492768595041322,1,d80-w12-h10.wav

2,-1,1,0.417192429022082,0.184917355371900,1,d80-w12-h20.wav

3,-1,1,0.355678233438485,0.027892561983471,1,d80-w12-h40.wav

4,-1,1,0.634069400630914,0.758264462809917,1,d80-w24-h10.wav

5,-1,1,0.607255520504731,0.192148760330578,1,d80-w24-h20.wav

6,-1,1,0.667981072555205,0.025826446280991,1,d80-w24-h40.wav

8,-1,1,0.039432176656151,0.022727272727272,0,small_bouncing_wooden_ball_1-pd.wav

9,-1,1,0.512618296529968,0.642561983471074,0,small_bouncing_wooden_ball_2-pd.wav

10,-1,1,0.169558359621451,0.081611570247933,0,small_bouncing_wooden_ball_3-pd.wav

11,-1,1,0.106466876971608,0.223140495867768,0,small_bouncing_wooden_ball_4-pd.wav

12,-1,1,0.439274447949526,0.758264462809917,0,small_bouncing_wooden_ball_5-pd.wav

13,-1,1,0.029179810725552,0.899793388429752,0,small_bouncing_wooden_ball_6-pd.wav

14,-1,1,0.753154574132492,0.754132231404958,1,w12_h10.wav

15,-1,1,0.809148264984227,0.390495867768595,1,w12_h20.wav

16,-1,1,0.849369085173501,0.851239669421487,1,w12_h40.wav

17,-1,1,0.812302839116719,0.628099173553719,1,w24_h10.wav

18,-1,1,0.876971608832807,0.291322314049586,1,w24_h20.wav

19,-1,1,0.934542586750788,0.033057851239669,1,w24_h40.wav

END SOb Sonic Browser - User Report File - 11-12-2002 05:06:29

Appendix J: Consent Form

Declaration of Informed Consent

I, the undersigned, hereby declare that I am willing to take part in a research project that is part of a Masters in Computer Science course assessment at the University of Limerick.

This study is entitled: ***Sonic Browsing and Sound Objects***
It is examining an application for one type of scenario, as a sound library tool, in terms of its usability, realism, and suitability for performing this type of scenario. This will involve showing users how to use the application (our prototype system), then performing a series of tasks and then asking questions to determine their views on the usability, realism and suitability for performing the tasks in respect the scenario.

I declare that I have been fully briefed on the nature of this study and my role in it and have been given the opportunity to ask questions before agreeing to participate. I understand that my role in this evaluation is as a co-evaluator and that this is not an evaluation of my ability, knowledge or intelligence.

I understand that after performing the tasks I will be asked to answer several questions in relation to usability, realism and suitability for performing these types of tasks, but that no personal, private or confidential information will be required of me.

I fully understand that there is no obligation on me to participate in this study and that I am free to withdraw my participation at any time without having to explain or give a reason. I am also entitled to full confidentiality in terms of the details of my participation and my personal details. I understand that some or all of the data (verbal and behavioural) may be used (quoted) in the report on the evaluation for illustrative purposes, but that I shall not be identifiable from this data either in the body of the report or in the appendices.

I also understand that my participation in this study may be recorded by video or audio means as well as notes taken by observers. I am entitled to copies of all recordings made during the session if I so wish to have them.

I acknowledge the fact that deception and concealment are inappropriate to, and not required in, this study, and that no attempt will be made to elicit information or actions from me using these means.

_____ _____

Signature of participant Date

Appendix K: Task Sheets

Sound Library Scenario

Task List for Evaluators - DataSet 1 - List 1

Evaluators Instructions:

Using our prototype system, please perform the tasks that are listed below, you may do so in any order. The observer will be present and will ask you to comment on your actions and reasoning for actions, this is known as the "Thinking Aloud" method. Please remember this is an evaluation of our prototype system, not an evaluation of you! Do not feel afraid to comment on the application either positively or negatively, as the goal of the experiment is to evaluate the application from a user's perspective.

Scenario:

You've been asked to compile a sound track for a major movie; you need to find the following elements to mix together to create the sound track.

Number of tasks to complete: 9 (nine)

Task List:

1. Find a sound of a toilet flushing and then tag & save them to the file 'toilet flushing'. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

2. Find a sound of seagulls and then tag & save them to the file 'seagulls'. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

3. Find any sounds of cats meowing and then tag & save them to the file ‘cats meowing’. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

4. Find a sound of a bugle and then tag & save them to the file ‘bugle’. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

5. Find any sounds of a ship’s foghorn and then tag & save them to the file ‘fog horns’. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

6. Find a sound of a small wooden ball bouncing three times and then tag & save it to the file ‘small wooden ball’. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

7. Find any sounds with the Environmental property ‘Car Passing’ and then tag & save them to the file ‘car passing’. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

8. Find any sounds with the Environmental property ‘Water Sounds’ **and** the Sound Source Property ‘Water’ and then tag & save them to the file ‘water sounds’. After saving the file, clear any tagged file/s by using

either User Settings -> Clear all tagged files or by using the mouse and keyboard.

9. Perform the previous task but select a single sound and using control (ctrl) – click to change its colour and then tag it & save the results to the file ‘changed colour’. After saving the file, clear any tagged file/s by using either User Settings -> Clear all tagged files or by using the mouse and keyboard.

Appendix L: Sound Designer Interview Questions

Questionnaire for Sound Designers on Sound Collections:

How many sounds would you work with in a complex sonic production?

How do you "navigate" your collection, when searching for sound to use?

Is this similar to what you do with other audio material, on CDs or other off-line collections such as tape or vinyl?

Do you use any computer tool or program to assist in the searching / browsing your sound collections?

If so, what do you use?

Can you highlight any particular features that you:

- find useful.
- think is problematic.
- think could be improved?

Do you use any classification or cataloguing tool for indexing or clustering within the sound collection?

What platforms/operating systems you use for sound design?

What software packages do you use for recording, synthesis and processing of sound?

Appendix M: Results from Questionnaires

Sound Library Scenario

User	Q1 - #Understand this system	Q2 - Learn to use this system	Q3 - To find a particular sound, when you know its filename	Q4 - To find a particular sound, when you don't know its filename	Q5 - To find a set of particular sounds with a specific property from a category
A	4	5	4	4	4
B	4	5	6	4	5
C	5	6	6	6	6
D	5	6	6	4	5
E	2	5	6	4	6
F	5	6	6	5	4
Min	2	5	4	4	4
Average	4.166666667	5.5	5.666666667	4.5	5
Max	5	6	6	6	6

User	Q6 - To find a range of sounds with any of a series of properties	Q7 - To find a particular sound when tagged	Q8 - To find a particular sound, when you knew its shape	Q9 - To perform an AND or an OR query using the filtering mechanism	Q10 - To change the details (Shape, Colour, Properties) of a particular sound
A	2			4	
B	4	3		4	
C	6	6		5	6
D	5	3		6	
E		6		6	4
F	6	2	5	4	4
Min	2	2	5	4	4
Average	4.6	4	5	4.833333333	4.666666667
Max	6	6	5	6	6

User	Q11 - To select a group of sounds	Q12 - To play the last sound that you listened to	Q13 - Overall, how would you rate the ease of use of this system	Q14 - How realistic did you find the sounds	Q15 - How did you find the quality of the sounds
A	3		4	5	6
B		6	4	6	5
C		6	5	3	3
D	5	6	6	5	5
E	6	6	4	6	6
F	6	3	5	5	5
Min	3	3	4	3	3
Average	5	5.4	4.666666667	5	5
Max	6	6	6	6	6

User	Q16 - Was there a noticeable play lag before a sound was played
A	3
B	6
C	5
D	6
E	0
F	5
Min	0
Average	4.166666667
Max	6