SOUNDNODE: A PROTOTYPE USER INTERFACE FOR SOUND DESIGN IN PERFORMANCE ENVIRONMENTS

Robert Andrew Stewart
Submitted in partial fulfilment of the requirements of the degree of Masters by Research (by creative work and dissertation)
July 2010
Faculty of the Victorian College of the Arts and Music
The University of Melbourne

Produced on archival quality paper
Abstract
This thesis delivers a functional software model for sound design that explores the ideas of agency, immersion and transformation in dramatic process in the context of human-computer interfaces. It seeks to examine the links between these ideas to develop a performance interface methodology that considers both the Neo-Aristotelian model for dramatic process and user interaction in its architecture. The validity of these ideas will be evaluated and explored through various prototype software systems I have designed and constructed within the Cycling ’74 Max/MSP/Jitter programming environment. Each will be extracted and discussed and usage examples will be examined to assess their viability and efficacy of the models. The final prototype is presented as a working software model for use as an interactive sound design and performance system.
Declaration

This is to certify that

(i) the thesis comprises only my original work towards the Masters except where indicated in the Preface*,
(ii) due acknowledgement has been made in the text to all other material used,
(iii) the thesis is 15000 words in length, inclusive of footnotes, but exclusive of tables, maps, bibliographies and appendices or the thesis is 13575 words as approved by the Graduate School, Faculty or RHD Committee.

Robert Andrew Stewart
Preface

Some of the prototype models demonstrated in this thesis were designed prior to the commencement of the research project. Where possible all prototypes are presented in chronological order and dated.

The software is designed within the Cycling ‘74 Max/MSP/Jitter environment. Whilst the software design and development is original the architecture that the software is built within is copyrighted to Cycling ‘74.

The software will be distributed under the Creative Commons ‘Attribution-NonCommercial 3.0 Unported’ licence. For more information on the particulars of this licence please visit: http://creativecommons.org/licenses/by-nc/3.0/legalcode

A section of the source code in the soundNode software has been adapted from Zac Poff’s Video Trigger software (http://www.zachpoff.com/software/video-trigger/) under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0) licence. The sections of code in question can be found in the ‘control’ sub-patch of the source code provided and were adapted for use in the ‘Space’ controller system.
Acknowledgements

I am indebted to the following people for their support and advice throughout the period of this research project. I would especially want to thank my supervisor and mentor Roger Alsop for all his encouragement and support.

Colleagues:

Olivia Crang Ilic, Jarod Factor, Alex Gibson, Katie-Jean Harding, Matthew Kneale, Dan Koerner, Tristan Meecham, Lily Paskas, Govin Ruben and Ellen Steele.

Companies:

The Hayloft Project, NYID (Not Yet It’s Difficult), Plot: Live, Spilt Second and x:Machine.

Special thanks:

Lance Hewett, Arlo Mountford, Emily Schinzig, Kate Sharkey, Annabel Sharkey, Shirley Stewart, Ranald Stewart, Peter Stewart, Rebecca Stewart, Chris Wenn and Zac Poff.
Table of Contents

Abstract ................................................................................................................................. 2
Declaration .............................................................................................................................. 3
Preface ................................................................................................................................. 4
Acknowledgements .............................................................................................................. 5
Table of Contents ................................................................................................................. 6
Table of Figures ................................................................................................................... 8
Table of Tables .................................................................................................................... 10
Chapter 1 - Introduction ......................................................................................................... 11
  1.1 Motivation .................................................................................................................... 11
  1.2 Overview of the thesis ............................................................................................... 12
Chapter 2 - Literature ........................................................................................................... 14
  2.1 Human computer interaction .................................................................................... 14
  2.2 Interaction design ...................................................................................................... 18
  2.3 Ubiquitous computing ............................................................................................... 22
  2.4 Computers as Theatre ............................................................................................... 23
  2.5 Current interface models .......................................................................................... 28
Chapter 3 - Discussion and analysis ...................................................................................... 35
  3.1 Introduction to soundNode ....................................................................................... 35
  Production environment ................................................................................................. 35
  Early (pre-research) prototypes ...................................................................................... 36
  Prototype 0 ..................................................................................................................... 39
  Prototype 1 ..................................................................................................................... 39
  Prototype 2: The current beta environment .................................................................. 42
  3.2 Describing the soundNode program ........................................................................ 43
  3.3 Constructing a sound design .................................................................................... 60
3.4 The software explained in terms of user interface principles ...............60
3.3 The software’s engagement with narrative........................................63
Chapter 4 - Conclusion .............................................................................64
  4.1 Conclusions ..................................................................................64
  4.2 Future directions ............................................................................65
Bibliography ...............................................................................................66
Index .........................................................................................................67
Appendices ...............................................................................................69
  A. Upgrade path ..................................................................................69
  B. Known issues ..................................................................................70
  C. Recommended system requirements ..............................................71
  D. Glossary ..........................................................................................72
Table of Figures

Figure 1 - Example of grid structure in a version of soundNode ........................................20
Figure 2 - The Neo-Aristotelian theory of drama .................................................................26
Figure 3. The Performance Triad Model(Sheridan et al., 2004) .......................................27
Figure 4 - QLab 2.3("QLab," 2010) ..................................................................................30
Figure 5 - Ableton Live 8("Live ", 2010) ..............................................................................31
Figure 6 - The Nintendo Wii™ system ..................................................................................32
Figure 7 - The Monome 40h ..............................................................................................33
Figure 8 - An early show control system ............................................................................36
Figure 9 - We Are What We Want performance system, which utilized live feedback from iPhone accelerometer data as a controller ......................................................38
Figure 10 - Prototype 0 ......................................................................................................39
Figure 11 - Prototype 1 v.1 ...............................................................................................40
Figure 12 - Prototype 1 v.2 ...............................................................................................41
Figure 13 - Prototype 1 v.3 ...............................................................................................41
Figure 14 - The default performance screen .......................................................................42
Figure 15 - soundNode program structure flowchart ..........................................................45
Figure 16 – The transport controls the selection and playback of scenes .......................46
Figure 17 - The follow-on panel allows the automatic re-triggering of scenes ...............47
Figure 18 - The Fader, Filter and Speed screens .................................................................49
Figure 19 - The Space screen features zone, proximity and scene jump triggers ..........50
Figure 20 – The Space screen provides a novel way to perform sound design concepts with the area of performance and receive feedback between the software and participants .............................................................................................................51
Figure 21 - Groups are assigned with a keyboard shortcut to correspond to one of four group trigger buttons on the left hand side. ..........................................................53
Figure 22 - Dynamic resizing of group trigger buttons minimizes mishits during use ........................................................................................................................................54
Figure 23 - The overview menu window provides realtime feedback of the state of the software during a performance ..............................................................................55
Figure 24 - The MIDI Learn window extends the affordances of the software by allowing external devices to control it’s functions ........................................................................56
Figure 25 - The trigger edit window allows users to program complex audio
triggers for later performance.................................................................57
Figure 26 - The program menu provides links to the utilitarian functions of the
software ........................................................................................................59
Table of Tables

Table 1 - Usability Heuristics of Nielsen and Molich ........................................... 15
Table 2 - Interface Design Elements by Brenda Laurel ........................................... 21
Table 3 – Janet Murray’s three categories for analysing interactive stories .......... 23
Table 4 - Brenda Laurel’s six qualitative elements in drama and in human-computer activity. ........................................................................................................... 24
Table 5 - soundNode Software Requirements Specification ............................. 43
Table 6 – Basic user interface principles (Talin, 1998) ............................................. 60
Table 7 - Known bugs or issues with the soundNode software ............................. 70
Chapter 1 - Introduction

1.1 Motivation

During my time as an undergraduate student of sound design in a school of performing arts one of the issues that I found hardest to reconcile and to negotiate was that of how technology fits into my artistic practice. It wasn’t so much that I had an issue with technology itself; I embraced any emerging forms or systems to complement and enhance my work and each example taken on their own merits they fulfilled their function. Often though many of the pieces of software or hardware that were commonly used in theatre, dance, puppetry etc. seemed in some way at odds with the creative process. Compact discs, Mini-discs, mixing desks, samplers, software show control systems all created some level of distance in regards to developing a narrative within the performance environment, either physically or mentally. Not only that but they demanded attention, focus that was pulled away from the text or action at hand. The rigid and often linear constraints these technologies were placing on the performance of sound design also did not seem to match the growing diversity and mediatization of performance art.

It was when I was introduced to Cycling ‘74’s Max/MSP/Jitter programming environment that I began to see the potential of computers to break with some of these conventions and give agency to the sound designer as performer. This research will aim to define the issues with these interfaces. It will then look at how users interface with dramatic process through the principles of Human Computer Interaction (HCI). A software model will then be developed and a working prototype presented as a system for creating and performing sound designs in a dramatic context.

The aim of this is to synthesize many forms of interaction from existing interface models into a single system that translates to performance practice. Its goal is to provide a system that is transparent and intuitive. It seeks to create a meta-narrative within the collaboration of the performance, one where the user agent’s actions are as important to the narrative as an actor or dancer. The model seeks to emphasise and enhance the duplex flow of communication between users, participants and observers. The system will capitalize on the digital computers
capacity for fluid interfaces and hyper-textual connexions. The interface will be translatable in its application to most forms of narrative. It is not a system for replacing the show control paradigm though. Its scope is to appeal to users who wish to perform and create the narrative of their sound design in the context of collaboration. It is concerned with not only stimulating communication between the user agent and the computer agent but also between performers, participants, observers and the model in a way that addresses the fluidity and immediacy of performance environments.

It will deliver a working software system for sound designers, artists and performance companies to design and perform sound in the same space and environment as the performance itself. Its value comes from mirroring not only the immediacy of performance but the fluidity to nuance a text or narrative in the way that an actor, dancer or musician would. It goes beyond this though by also providing a system that can provide a hyper-textual methodology for creativity and expression of narrative and audio design.

1.2 Overview of the thesis

The remainder of the thesis is organised into three chapters.

Chapter 2
In the second chapter I look at literature and history surrounding the development of HCI, the rise of ubiquitous computing, the notion of computers as theatre and interaction design. At the conclusion of this chapter I have defined the structure of the software model that is developed and discussed in Chapter 3.

Chapter 3
Chapter 3 discusses and analyses the evolution of the software system I have developed. Early prototypes are briefly overviewed to demonstrate history and motivation for the final three prototypes. Prototype 0 and 1 are described and critiqued according to the dramatic and interface structures outlined in Chapter 1.
Prototype 2 is then thoroughly broken down to each element and its viability and efficacy is assessed.

**Chapter 4**

The thesis is concluded and the successes and failures of the final prototype are deliberated. Scope for future work is discussed and the deployment of the model for practical application.
Chapter 2 - Literature

Before the sound design software system can be developed the form and function of it must be defined. In this chapter I will look at Human Computer Interaction (HCI), Ubiquitous computing, Interaction design (IxD), Computers as Theatre and provide an overview of current interface models relevant to this research. These topics form the basis for the methodologies referenced and employed in Chapter 3 with the prototypes.

The section on HCI will explore the influence usability heuristics, such as those developed by Jakob Nielsen, have had on the development of software models. Interaction design will look at the work of Donald Norman and Brenda Laurel in the field of interaction design and breakdown interface elements, design elements and look at the ideas of affordances and personas in the design process. Computers as Theatre will provide an overview of the work of Brenda Laurel, Janet Murray and Michael Mateus and look at their integration of Aristotelian process into user interface and experience design. Current software and hardware models relevant to this research will then be discussed before formulating a conceptual model for the design phase of the research.

2.1 Human computer interaction

*First Law - A computer shall not harm your work or, through inactivity, allow your work to come to harm.*

*Second Law - A computer shall not waste your time or require you to do more work than is strictly necessary*

*Jef Raskin (Raskin, 2000)*

Human Computer Interaction, or HCI, is the study of the principles that define the communication between users and computers. Pioneers of the field, such as Jakob Neilsen, Rolf Molich, Donald Norman and Jef Raskin, strived to not only understand the processes underlying the communication between humans and technology but to improve them.
2.1.1 Usability heuristics

The following are ten general principles for user interface design. They are called "heuristics" because they are more in the nature of rules of thumb than specific usability guidelines. Jakob Nielsen and Rolf Molich originally developed ten usability heuristics (Nielsen, 1994) as not only a guide to evaluate software usability but to speed up the process of this evaluation. Adhering to these heuristics in combination with inquiry, inspection and testing will strengthen the mediation of the performance through the software. The interface, as a working presentation of the derived taxonomic system assists in understanding dramaturgical relationships between intention and action. An impeded relationship between the two is detrimental to the flow of the performance.

<table>
<thead>
<tr>
<th>USABILITY HEURISTICS OF NIELSEN AND MOLICH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visibility of system status</td>
</tr>
<tr>
<td>Match between system and the real world</td>
</tr>
<tr>
<td>User control and freedom</td>
</tr>
<tr>
<td>Consistency and standards</td>
</tr>
<tr>
<td>Error prevention</td>
</tr>
<tr>
<td>Recognition rather than recall</td>
</tr>
<tr>
<td>Flexibility and efficiency of use</td>
</tr>
<tr>
<td>Aesthetic and minimalist design</td>
</tr>
<tr>
<td>Help users recognize, diagnose, and recover from errors</td>
</tr>
<tr>
<td>Help and documentation</td>
</tr>
</tbody>
</table>

Cognitive Load Theory (CLT) is also a very important consideration in the design methodology. According to Brian Chipperfield, CLT is based on the following principles of cognitive learning (Chipperfield, 2006):

1. That short-term memory (working memory) is limited in capacity to about seven informational units.
2. Long-term memory is unlimited in capacity and is where all information and knowledge is stored.

3. Knowledge is stored in long-term memory as schemas or schemata.

4. Schemas, no matter how large or how complex, are treated as a single entity in working memory.

5. Schemas can become automated.

Sensory memory is the most volatile. Visual information can extinguish within half a second and auditory information within three seconds if not identified, classified and assigned meaning too. Short-term memory, or working memory, can hold around seven distinct pieces of information to be processed give or take one to two on either side. These ideas are essential to the makeup of a software interface and its level of usability. Interfaces and modes of interaction with a high associated cognitive load can become harder to use and learn (pass on to long term memory). High cognitive loads also detriment the user’s ability to respond and fulfil tasks because of the limitations on working memory.

Some general issues with audio software and hardware have been detailed below:

2.1.2 Issues with audio performance software interfaces

One of the major issues with software interfaces is their dependency on visual feedback. Visual feedback is often more necessary for less experienced performers whereas more experienced performers rely more heavily on what they are hearing and on the interactions of the software with the performance. A certain level of trust can be built between an experienced user and a familiar interface because of the certainty or predictability of feedback arising from using the interface regularly.

Screens and touch interfaces can sometimes be unsuitable or impractical for performance. They can obscure and draw focus from the operator’s view of the piece. Issues can also arise if the performance is situated in more than one area and confusion from could exponentially increase if a users attention is drawn between these spaces and heavy interaction and feedback demands by a particular piece of software. There are much more important visual elements for a user to concentrate on in a sound design performance such as interaction
between performers and with the user, interaction with the audience, the reading of musical scores and dramatic cue sheets and instructions.

Software interfaces to the untrained or uninitiated user often require extensive training and special or intimate knowledge with particular technical languages and concepts before they can be effectively used. Another major consideration, depending on the hardware device peripherals associated with a particular piece of software, can be that of the lack of tactile and haptic feedback. This is especially significant in the case of touch screens.

2.1.3 Issues with audio performance hardware interfaces

A major issue and consideration with hardware interfaces is the responsiveness and latency of physical actions and their outcomes. Users pressing a button ideally expect instant feedback. Any large amount of latency, even a measure of milliseconds, can throw off the users connection with their own actions and that of the software.

Simple and effective layouts of the controller elements are also paramount in hardware interface efficacy. A certain level of rigorous practice and familiarity is required to achieve expected and reproducible results. A well-designed hardware interface will reduce the amount of time it takes to build muscle memory for effective interaction through practice with the device.

The cost of hardware interfaces must also be considered. Sometimes the cost of buying or hiring hardware can put unrealistic constraints on the performance project. There are also costs involved with having a separate operator present to perform the sound design as opposed to a stage manager who operates sound, lighting and AV from a single automated console.

Space restrictions and hardware size often determine the use and approach to hardware. Quite often competing space is a factor in operating areas. Restricted movement can result in restricted interaction.

An understanding of the principles of HCI is fundamental to the construction of user interfaces. It provides the language for communication between the user
agent and the technology so that their intentions, or intended use of the interface, can be realised into action.

2.2 Interaction design

“Interactivity is an increase in a reader’s participation. It’s a bidirectional communication conduit. It’s a response. Interaction is a relationship. It’s mutually executed change. It’s indeterminate behaviour, and the redundant result. As far as narrative is concerned it amounts to providing the reader with the ability to alter specifics in the plot.” Mark Stephen Meadows (Meadows, 2002)

Mark Stephen Meadows in his book Pause and Effect: The Art of Interactive Narrative Interactivity is a duplex flow of information to communicate ideas and requires rules and constraints to function effectively. Interaction implies a modification. A change to something that already exists or is established. Meadows defines three core principles of interaction and four steps in the process of interaction:

Three principles of interaction:

1. input / output
2. inside / outside
3. open / closed

These binaries are extremely useful when defining the overall system topology. Input and output are reciprocal; input creates output and vice-versa. Inside and outside can be thought of in terms of users and of the interactive system. An open system is one that has unfamiliar or unpredictable outcomes whereas a closed system has variables which a set and predictable.

The four steps of interaction:

1. Observation
2. Exploration
3. Modification
4. Reciprocal Change
These four steps quantify for interface designers the stages of interaction that are used not only in development but also in evaluation of the quality of engagement and communication.

*All art is an interaction between the viewer and the artwork, and thus all artworks are interactive in the sense that a negotiation or confrontation takes place between the beholder and the beheld.* (Dixon, 2007)

### 2.2.1 Grid Systems

“Designers can seldom go wrong starting with a grid system. A grid system helps designers organize information into a coherent pattern. [...] Grid systems aren’t just designs on graph paper; done well, they help structure screens so that there is a clear visual hierarchy and flow through the elements on the screen.” (Laurel, 2003)

As Chipperfield has demonstrated overloading users cognitively can have adverse effects on how they interpret and process information. Grids are a strong aesthetic form with a basis in art and mathematics that help organise information and communication. Below is an example of how a grid structure is used in the design of the soundNode software.
2.2.2 Visual Flow

In the Western world, the eye generally travels from left to right, top to bottom, and
designers should be aware of this flow and design for it. Don’t force users’ eyes to jump all
over the screen. (Saffer, 2007)

The grid paradigm defers to the symbolism of the drum machine, the keypad and
the MIDI controller. After decades of saturation from these devices the grid
structure is immediately familiar. This could even be considered to be comforting
in terms of its interactive potential.

Interfaces that have well implemented visual flow allow for stronger and more
elegant communication of ideas and information.
2.2.3 Visual Elements

“The most engaging interactive narrative relies upon flow; that is, uninterrupted participation in the unfolding action. Poor interaction design can interrupt flow and degrade the experience.” (Laurel, 1993)

Visual elements are the defining characteristics of any graphical user interface (GUI). The metaphors they employ and their familiarity and ease of use set the constraints on how the user can interact with the system. Brenda Laurel defined a table of most common design elements found in interfaces.

Table 2 - Interface Design Elements by Brenda Laurel

<table>
<thead>
<tr>
<th>Interface Design Elements</th>
<th>Physical only</th>
<th>Digital only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common to Physical and Digital interfaces</td>
<td>Jog dial</td>
<td>Checkbox/radio button</td>
</tr>
<tr>
<td>Buttons</td>
<td>Joystick</td>
<td>Twists</td>
</tr>
<tr>
<td>Dials</td>
<td>Shuttle</td>
<td>Dropdown menu</td>
</tr>
<tr>
<td>Latches</td>
<td></td>
<td>Multi-select list (list box)</td>
</tr>
<tr>
<td>Sliders</td>
<td></td>
<td>Spin box (increment)</td>
</tr>
<tr>
<td>Handles (edges/corners)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Most elements of the above table are utilized extensively in everyday software interfaces and are immediately familiar to the average user. The table could be expanded to include gestural and wireless controllers and multi-touch control surfaces. While using these elements in interface designs is not a guarantee of user familiarity with their nature and function, they symbolise the strongest and most dominant forms of communication metaphors between users and interfaces.

2.2.4 Affordances

"...the term affordance refers to the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used. [...] Affordances provide strong clues to the operations of things. Plates are for pushing."
Knobs are for turning. Slots are for inserting things into. Balls are for throwing or bouncing. When affordances are taken advantage of, the user knows what to do just by looking: no picture, label, or instruction needed." (Norman, 1988)

Affordances are the scope of possible interaction available to the user through the design of the interface. Donald Norman extended the definition of the idea of affordances, which was first put forward by psychologist James J. Gibson in 1977 (Gibson, 1977), to include not only the actual affordance offered my an interface properties but also the perceived affordance.

2.3 Ubiquitous computing

For thirty years most interface design, and most computer design, has been headed down the path of the "dramatic" machine. Its highest ideal is to make a computer so exciting, so wonderful, so interesting, that we never want to be without it. A less-traveled path I call the "invisible"; its highest ideal is to make a computer so imbedded, so fitting, so natural, that we use it without even thinking about it.”(Weiser, 1988)

Regarded as the third wave of computing, the notion of ubiquitous computing is changing the way we think about computer systems and interfaces. The first wave was that of the Mainframe, one computer for many people. The second wave was that of the personal computer (PC) which was of once computer for each person. The third wave is that of ubiquity, many computers for a single person.

Computers are now everywhere and can be enlisted to aid us in the smallest or largest tasks. As a society we accept, wittingly or unwittingly, that computers may be inside the most trivial of devices and appliances.
2.4 Computers as Theatre

In her 1998 book ‘Hamlet On The Holodeck’ Janet Murray proposed three categories for analysing of interactive stories as experienced by a user/participants: immersion, agency, and transformation.

<table>
<thead>
<tr>
<th>Immersion</th>
<th>The feeling of being present in another place and engaged in the action therein. (Suspension of disbelief)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agency</td>
<td>The feeling of empowerment that comes from being able to take actions in the world whose effects relate to the player’s intention. Agency over the interface and performance are at the forefront of concerns of the soundNode software interface. Not only agency over the manipulation and delivery of sound to the performance environment but power over the narrative its affect on the performance system.</td>
</tr>
<tr>
<td>Transformation</td>
<td>According to Michael Mateus, Murray’s idea of Transformation can have three interpretations. Transformation as masquerade, transformation as variety and personal transformation. (Mateas, 2001)</td>
</tr>
</tbody>
</table>

Here is a summation of Michael Mateus’ interpretations of the concept of Transformation in interactive narrative.

*Transformation as masquerade:* The game experience allows the player to transform himself or herself into someone else for the duration of the experience. This can be measured in the integration of the performer into the performance. Stage managers and sound operators have never sat outside of the performance process, they are as much characters and agents in the narrative as are the actors, dancers, musicians or audience members. As with the Aristotelian model...
for dramatic process there is a focus reiterated here on intensification not extensification. The characteristics of the performer and the nuanced and fluid integration are amplified by the stress placed on an improvised and responsive dialogue between the sound design and the performance narrative.

Transformation as variety: The game experience offers a multitude of variations on a theme. The player is able to exhaustively explore these variations and thus gain an understanding of the theme. By keeping every element of a scene as a separate entity or part, the soundNode model can provide range, variation and the ability to add and edit its position within the narrative on the fly. This allows for a creative process that can be extended and developed into a performance process much like with jazz musicians or DJ’s.

Personal transformation: The game experience takes the player on a journey of personal transformation. This can be either explicit or implicit. It is important to be reminded of this though as it relates to classic narrative. Although narrative structure may always be changing the emphasis on experience for performers and participants is still the focus of live performance.

Around the same time Brenda Laurel presented a new perspective on the qualitative elements of Aristotelian dramatic process in terms of HCI in her influential book Computers as Theatre (Laurel, 1993). Table 3 outlines her view of how these elements translate to the interaction design. This outlook has provided us with not only a method for designing interfaces but also a framework for how they fit back into drama.

Table 4 - Brenda Laurel's six qualitative elements in drama and in human-computer activity.

<table>
<thead>
<tr>
<th>ELEMENT</th>
<th>IN DRAMA</th>
<th>IN HUMAN COMPUTER INTERACTIVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>The whole action being represented. The action is theoretically the same in every performance.</td>
<td>The whole action, as it is collaboratively shaped by system and user. The action may vary in each interactive session.</td>
</tr>
<tr>
<td>Character</td>
<td>Bundles of predispositions and traits, inferred from agents’ patterns of choice.</td>
<td>The same as in drama, but including agents of both human and computer origin.</td>
</tr>
<tr>
<td>Thought</td>
<td>Inferred internal processes leading to choice: cognition, emotion, and</td>
<td>The same as in drama, but including processes of both human</td>
</tr>
</tbody>
</table>
Michael Mateus in his paper *A Preliminary Poetics* (Mateas, 2001) then extended the model to incorporate interactive forms of narrative, such as computer games. His model has three significant additions. He introduces material for action and inferred formal cause as parallel quantitative elements that bear influence on interactive narrative and user action as an integrated qualitative consideration on character in the narrative.

Materials for action are the tools that are available to users to affect change on over the narrative. The inferred formal cause is the users own perspective of the narrative and the direction they wish it to take. Combined these elements integrate the user within the narrative and they then become a character in the story.

It is useful and pertinent to begin assessing the crossover between computer games and performance interfaces. Realtime feedback in software and diverse methods of non-linear communication between agents affect game-like constraints on the narrative system previously little realized in performance. Narrative elements can be effectively played with some forms of technology, and the outcome is then determined by the state of play as negotiated by the participants. There are also practical similarities in spatial audio and lighting triggers and control and from the diverse forms of data that can be fed back to users through the physical performance environment. Brenda Laurel has modelled the effect of this user action on narrative as seen below in Figure 2.
In 1993 Brenda Laurel wrote her influential work “Computers as Theatre” and significantly changed the way people not only thought about designing computer software and interfaces but also how technology is integrated into creative practice.

The user and the technology are both individual agents working together towards a performance outcome and a hybrid agent. With integration and practice, as with musical instruments, technological interfaces can become a cybernetic extension of the performer. It is not simply a case of prostheses though but symbiosis. It is also not a commensal relationship either where one agent benefits and the other gains nothing. Both the technology and the user evolve and co-exist, mutually benefiting each other. This results in a state of flux in the sentiment and direction of both interest in aspects of technologies and interfaces and the directions of their
development. The design demands and constraints of one element directly affect the development of the other.

Jennifer G. Sheridan, Alan Dix, Simon Lock and Alice Bayliss, whilst developing the Performance Triad Model (PTM) (Sheridan, Dix, Lock, & Bayliss, 2004), put forth that:

“Performance imposes its own aesthetic and practical constraints, but these are different from those most often associated with computing research. In computing, the demand for face validity, an implied potential utility, restricts the freedom with which we can explore particularly the new forms of interaction afforded by technology.” (Sheridan et al., 2004)

The Performance Triad Model describes the relationship between the performer, observer and participant, as mediated by technology, and within the bounds of context and environment (see Figure 3). They continued by stating that this relationship is a break from the original emphasis on efficiency in HCI and necessitated the need for a clearer definition of performance so that a critical dialogue could be engaged.
The overview of this definition stated that “live performance may be seen as “tripartite interaction”, i.e., synchronous, real-time interaction that occurs between a number of agents (performers, participants and observers) implicated in a performance.” (Dix, Sheridan, Reeves, Benford, & O’Malley, 2005) The performers construct the ‘frame’ of the performance, and perform to an audience with a particular (typically premeditated) intent. The participants interpret the ‘framing’ of the performance, and are engaged in the intended performance through some witting or unwitting activity. The observers also interpret this performance framing and although their presence is integral to the performance, their performative actions are not. They may also interpret a framing either wittingly or unwittingly.

The formulation of this model as the basis for the way performance is mediated through technology provides both researchers and practitioners with a common dialogue. This framework is a platform that enables a discourse on specific elements of either the technology or performance, in this case the software/user interface.

Furthermore taking these models under consideration we can see how introducing interactive frameworks for communication into software and hardware models can alter the idea of what a dramatic performance, dance or installation might be. We begin to see elements of computer games, hypertext and generative scenarios permeate the structure of the narrative.

### 2.5 Current interface models

Since Charlie Richmond of Richmond Sound Design (RSD) developed the first computerised audio control system for performance in 1986 (COMMAND/CUE™) the focus of most theatrical audio systems has been on providing the user with extended control and automation of the parameters affecting the sound design. In 1991 Richmond extended this control system to other aspects of theatrical performance with *Stage Manager™*, a system that used MIDI Show Control (MSC) to automate and control lighting, sound, lasers and rigging from a single system.
These systems are largely reactive and not interactive though. Users trigger cascades of sound events at rigidly defined points in a pre-determined text. The systems can be defined as closed systems. The output from the users input is highly predictable and controlled. These are qualities that are necessary for some forms of performance systems though. Where cues are highly complex and time critical, beyond the scope of a single operators capabilities to reproduce on call.

Since the development of show control systems software and hardware for audio have developed to accommodate an increasingly diverse range of performance practices. Below I give four examples of different systems for performing sound. Two are software and two are hardware. Figure 53’s QLab is an example of a popular form of the show control paradigm. Ableton Live is a music performance system that offers an open solution for controlling sound in a non-linear manner. The Nintendo Wii is a popular gestural controller and finally the Monome is a simple but powerful OSC hardware controller.
QLab is an example of a very popular theatrical show controller made by the company Figure 53. It is quite heavily based on the original show control model developed by Charlie Richmond. It can control sound, video and MIDI through the use of a linear cue list. Cue lists are organised into events in time where multiple layers of audio and visual information can be activated in sequence.

Whilst it has the capacity for triggering cues and controlling aspects, such as volume and auxiliary send levels, of the audio through the implementation of MIDI, MSC and MIDI SysEx (System Exclusive) messages the interaction between the user and interface can be classified as reactive and not interactive. Events are activated through user input. The user then gets feedback through the interface and the output of the software. Then the software waits for further input.

QLab currently represents one of the most accessible software paradigms for theatrical show control, both through price and its functional design. The emphasis
is however is on the extensive, and lengthy, pre-mapping of events so that the sound design being presented can be faithfully reproduced from performance to performance.

**Ableton Live**

![Ableton Live](image)

*Ableton Live* is a loop-based software music sequencer and Digital Audio Workstation (DAW). Unlike QLab the emphasis of the software’s design is on altering and manipulating the performance of a grid of synced loops and one-shot samples. It has two main views, the arrangement and session screens. The arrangement view is primarily for organising and performing clips, loops or sections of sound that are defined in the session view. The session view is for recording and editing sounds and follows the established software paradigm of the DAW that is in turn derived from the recording studio.
Ableton Live’s popularity as an audio software paradigm comes from its ability to map and alter parameters with seamless fluidity while performing. It has extensive capacity for altering the music or sound being played back but is primarily a tool for music production and live performance. The primary focus of its design is for a single operator to communicate a composition directly to an audience though the addition of Max For Live, a modular extension for integrating Max/MSP/Jitter programming into Live, has greatly increased the scope of its architecture for utilising a large range of input and output controller data.

*Figure 6 - The Nintendo Wii™ system*

Nintendo’s Wii™ gaming system has had a revolutionary impact on the way everyday users look at interaction. The wireless gestural control system is relatively cheap and because it uses the Bluetooth wireless communication protocol to transfer data the spatial coordinates can be harvested by other software and used for performance.

It has been used extensively as an input and performance device for theatrical performance and installation works since its release because the expressive
capacity of the device has captured the imaginations of many users. It provides a way of translating the language encoded in movement and gestures into the realm of the virtual.

*The Monome*

![The Monome 40h](image)

*Figure 7 - The Monome 40h*

The 40h has no labels on the unit itself. At first, this may seem confusing or intimidating, but after spending a short time with the unit you will realize that this ambiguity is one of its major strengths. The lack of labels leaves the user with an open palette to choose from. (Dunne, 2007)

As suggested by the quote above from Jared Dunne, the Monome is an excellent example of interface design for performance. What the minimalist design of the Monome units has to offer is in both exciting the imagination of the users, by minimising cognitive load and by maximizing its application. Its potential is enormous but not prescriptive.
Users can assign meaning to the interface by the patterns and actions they ascribe to it. At its simplest the Monome can accommodate 64 linear cues or triggers. But the interface could become so much more. It could take the form of a spatial metaphor, sounds and triggers arranged in terms of their relationship to the performance space. Or it can used to form abstract patterns or symbols that bear a metaphorical relationship to the work.

The software and hardware examples above provide a cross section of some of the most popular methods for performing sound.

### 2.2.5 Towards a conceptual model

There are many examples of hardware, software and hybrid solutions for audio performance and sound design. From the most prescribed and structured models of cue-to-cue playback to the most open ended and unconstrained methods of generating and triggering sounds. To build an interface that synthesises some of the most effective, simple and powerful aspects of these examples we must first extract elements of them and assemble them into working models to test their efficacy.

The next chapter will discuss a series of prototype software models produced for this thesis. With the principles of HCI and IxD in mind, and with the influence of ubiquitous computing, the Neo-Aristotelian model of interactive narrative and a base of theatrical and music performance software and hardware as a reference point, a conceptual model for an interactive audio performance interface will be put forward.
Chapter 3 - Discussion and analysis

3.1 Introduction to soundNode
The model that became the final system for this research affords the user the opportunity to assemble a design through play and interaction with the other participants of the performance. The control surface is also the design surface. It allows the user to quickly ‘drag’ sounds from the desktop or a folder and begin to use them to interact with the performance. This is the most simplistic and high-level use of the software. More complex interactions follow as the design narrative begins to work with the performance discourse. A user can manually retrigger the sound; loop or sequence the sound to a global tempo; explore the individual events of a scene using a spatial metaphor; use input from MIDI or sound from the space to trigger events. The focus of the interface design is not only about reaction but interaction, of creating a dialogue between users, the system and performers where the communication is simple, powerful and intuitive.

Production environment
One of the powerful traits of the Cycling ’74™ Max/MSP/Jitter environment is that rapid prototyping can be achieved within the program itself. Because the default UI objects are available to the programmer within the editing environment, paper prototyping (something that is usually tackled outside of the coding environment) can be worked on in real time. Whilst this does pose some issues for the designer/coder in getting distracted by the particulars of the design and by starting to code working UI objects, it does afford a design environment where the ideas generated are a direct reflection of the possibilities and constraints of the coding environment.

I have separated the prototypes I am discussing here into Early prototypes, Prototype 0, Prototype 1 and Prototype 2. The Early Prototypes were not all designed before the other three prototypes. Some are experimental versions which where workshopped and rejected from various stages of the design phases.
I will look briefly at the evolution of these prototypes and how it impacted on the final, or current, working beta environment.

**Early (pre-research) prototypes**

![An early show control system](image)

**Figure 8 - An early show control system**

There were many early prototypes. One of the earliest functional prototypes, which was used in the sound design of various shows (Spring Awakening by the Hayloft Project, Senseless by x:Machine, The Kids Can Get Lost by Spilt Second) was fairly rudimentary (see Figure 8 - An early show control system). It came out of my frustration at the time of using Compact Disc or Mini Disc players in combination with mixing desks to realise sound designs. Designing for performance through the technology of CD’s and MD’s were cumbersome and disruptive to the performance environment, often necessitating the need to be outside of the rehearsal process than in it. Updates to single sounds that had been tried and rejected from the design often meant a trip home to the desktop computer to edit and burn a new CD, often only to have the process repeated.
As computers became more accessible, portable and powerful various theatrical audio software systems began to become available to a greater audience of performance artists. It is interesting to note though that many theatres and performance venues still use CD and MD players for shows, whether out of habit, budgetary constraints or a perceived reliability offered by the familiar.

The prototype below, used in the installation ‘We Are What We Want’ by Alex Gibson differed from other early models in that it had tempo-sequenced events. The live playback of the events and effects were controlled by an Apple iPhone™ hidden in a prop. The iPhone was sending accelerometer data via a wireless network connection to the software. Whilst audience members were immersed in the performance, interaction with the prop gave them agency over the sound design and therefore over the performance itself.

The major attraction of this model as a performance tool was the bi-directional flow of communication and interaction with the audience/performers and the designer/performer. The new layer of possibilities and reflex provided by the interface made the interface exciting and engaging for observers, participants and performers. It provided an open system that was unpredictable but was constrained within the rules and context of the sound files and effects chosen by the user.

It did however rely heavily on one type of input, a stream of accelerometer data being sent via Open Sound Control (OSC) over a wireless network from and Apple iPhone™. As such it lacked the capacity as a system model to be applied to the variety of performance narratives that I was aiming to cover with my research prototype.
Figure 9 - We Are What We Want performance system, which utilized live feedback from iPhone accelerometer data as a controller
Prototype 0 was a reconditioning of the earliest functioning controller. Its interface was organised in a cleaner and more functional grid formation and it had a strong visual task flow from left to right and top to bottom. Its major strength was that it achieved all of its function within a single screen interface. Within relatively short periods of time users could become familiar with the mechanics of interacting with the software and could begin adding sounds responding to cues from the performance.

Its weakness as a system was that it was still very limited to the linear playback of cues in a sequence with only minimal manual control via volume and channel routing that could be affected during that playback.

Prototype 1

The Prototype 1 phase had three main versions before being abandoned. All three were essentially more complex versions of the earliest prototype. Their interfaces varied slightly but all were in their simplest forms multi-channel show controllers.
These versions did introduce some advances in the interface and programming. Switching spaces within a single screen via tabs became a successful method in my trials of managing a more complex array of options without inundating the user with cascading windows and pop-ups.

MIDI control, effects filters, speed and pitch parameters, cue/scene management and input triggers were all trialled and found to be useful contributions in the creation and performance of sound designs.

![Prototype 1 v.1](image)

Figure 11 - Prototype 1 v.1
No matter how the interface was developed though it lacked the agency that was required to make it suitably interactive and responsive for performing. After seeing a video musician performing on a Monome I decided to test the grid performance.
structure to see if it would be compatible with the rest of the show controller architecture I had already developed. The result was Prototype 2.

Prototype 2: The current beta environment

Figure 14 - The default performance screen

Overview of the soundNode software

This is the latest iteration of the software controller. It demonstrates a shift in the metaphors employed for interaction. It is a multi-channel (eight in and eight outs) sound and MIDI performance system.
### 3.2 Describing the soundNode program

I will first outline the starting point for the design in the form of a Software Requirements Specification (SRS) table. It is a well-established procedure in software engineering for a SRS to be researched and generated at the beginning of a project to outline the design considerations and stakeholders involved. In this case it will provide an overview of the consideration made in the design of the final prototype before discussing the more specific aspects of its form.

#### Table 5 - soundNode Software Requirements Specification

<table>
<thead>
<tr>
<th>Overall Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Program perspective</strong></td>
</tr>
<tr>
<td><strong>Program functions</strong></td>
</tr>
<tr>
<td><strong>User characteristics</strong></td>
</tr>
</tbody>
</table>
computer and audio hardware systems and be able to operate them.

**Constraints**

The processing power of the software will be limited to that of users system and hardware. User experience will vary according to the hardware controllers (if any) employed by the user.

**Assumptions and dependencies**

The program is reliant on a hardware system capable of running it as per the minimum specifications.

### Specific requirements

**External interfaces**

Audio hardware interface (internal or external sound card) is not required but desired to realise the full potential of the software. MIDI and USB devices can be utilised as input controllers for the software.

**Functional requirements**

Sound file need to be loaded and played back quickly and intuitively. The interface should provide a variety of options to the user for affecting the play back of these sound files. The software must be able to create, load and save different sets of these sound files.

**Performance requirements**

The program should facilitate a high fidelity multi-channel sound design that is free from audio artefacts and latency issues.

**Logical database requirements**

Ability to save, load, edit and archive ‘shows.’

**Design constraints (non-functional requirements)**

The system is intuitive and accessible. Users can determine and the level of interaction via the interface at
Software system attributes

Can be utilised by a variety of platforms and systems. Can be employed ‘low end’ systems. Not constrained to heavy, high specifications or external hardware dependencies.

I will now discuss the design and function of the different components of the interface that were the product of the constraints of the SRS. The components are the Transport, the follow-on panel, the five performance screens, group triggers, the overview screen, the trigger edit screens and the program menu.

The basic overall structure of the program is outlined in the flowchart below. Following that there is an in-depth breakdown of each of the components:

Figure 15 - soundNode program structure flowchart
The transport controls the selection and playback of scenes. A scene is a palette of up to twenty-four sounds and/or events that can be further organised into sub-groups. Like scenes or acts they have parts of the narrative that can be performed together or separately. Whilst they are not necessarily linear they embody the essence of narrative and dramatic structure.

The ‘GO’ button in this case is a subverted reference to other sound and show control systems, such as Figure 53’s QLab™, Cricket v2™ or Stage Research’s SFX™. Whilst navigating it gives feedback to the user by turning green for the current scene that is loaded and orange for any other cue that has yet to be instantiated. Hitting ‘GO’ sets the new scene, loading any changes to sound file or input assignments or MIDI events. This makes them available to the user to then be triggered. If any of these events are set to be ‘cued’ (see The trigger edit windows) then hitting ‘GO’ will trigger the cued events. The ‘GO’ button can also be activated by hitting the space bar on the computer’s keyboard.

Using the pause/resume button can halt all sound file play back. This is in addition to the pause/resume buttons on single triggers.

The ‘next’ and ‘prev’ buttons navigate forward and backward through the scene list. The name of the current scene is displayed to the right. Scenes can be loaded in any order the user wishes.

The narrow bar button beneath the transport collapses the main screen to only show the transport. This can be employed for either economising screen real estate when viewing other windows, such as the edit windows, or for the simple operation of cued only scenes that require no other user performance input.
The Follow-on panel

Figure 17 - The follow-on panel allows the automatic re-triggering of scenes

The follow-on cue section allows the automatic retriggering of the next scene in the scene list after either a designated period of time or after all other fades and cued events have concluded. It affords complex sequences of scenes to be played back concurrently.

The Performance screens

The Triggers and Space are tools designed for direct engagement with the performance narrative structure. Faders, filters and speed affect the nature of the elements. They are also performance tools but are of a secondary order of interaction. All the performance screen’s tools were chosen for users to have, not only a set of tools useful for sound design, but tools that had distinct agency over the performance of the sound.

The screens are accessed via labelled tabs in the top left corner of the performance interface. They can also be switched between by using the keyboard numbers one through to five.

The trigger pads represent an ordered grid of decisions that can be made in the context of the scene from the perspective of the sound performer/designer. The Space tool affords a more fluid and gestural system for exploring the delivery of the narrative elements. The faders, filters and speed controls provide the user with a way of manipulating the elements during their delivery into the environment of the performance narrative.

Below I look at each of the performance screens and how they function.

Trigger pads

The trigger pad screen is selected by default when the software is loaded. It is a palette of narrative components, providing agency over the both the interaction
and the performance. The trigger pads are a grid of blank buttons. A tabula rasa waiting to be filled with narrative constructs and organised into systems fornuancing performance.

The interface has been designed to affect action and participation in the narrative of the performance. The pads highlight when the mouse glides over them, they invite a response from the user, and they suggest output from the users input. The group triggers to the left of the pads are symbols of action. They are used to form relationships between the narrative elements in time. Patterns and sequences can be set in motion and repeated. Combinations of sounds can be interjected into the performance to create further complexity.

The task flow in this case sits hand in hand with the narrative flow. It is not linear though. Sections of the narrative can be arranged and grouped to form relationships. They are not however tied to these relationships. They are not even tied to the organisation in itself so far that the sounds can be selected and reselected, combined and recombined in endless permutations.

Sounds can be dragged from the desktop or a folder and dropped onto a trigger. Hitting the trigger will then play the sound. This provides a direct analogy of the inside/outside dichotomy of not only the system interface, but of the performance environment. Sounds are literally dragged into the narrative from the outside world.

**The Fader, Filter and Speed screens**
The Fader, Filter and Speed screens are used to manipulate the quality of the sound elements that are triggered. These screens employ visual concepts that are closely related to mixing desks and DAW’s.
Figure 18 - The Fader, Filter and Speed screens

The Fader screen can manually adjust the volumes or velocities of individual audio triggers or monitor the progress of automated changes. Levels that are automated can easily be manually adjusted after the course of the programmed fade has run.

The Filter screen is a set of single band equalizers for manipulating and adjust the frequency spectrum of the audio playing. It only applies to the play back of sound files and not MIDI events.

The Speed dials or pots have a 270 degree rotation and alter not only speed but the pitch of the sound file being played back. This can be used to manipulate and affect the quality of the sound being played but also how it fits temporally into the text. Speed and pitch are altered and with it our perception of time.
The Space screen

Figure 19 - The Space screen features zone, proximity and scene jump triggers

Clicking the ‘SPACE’ tab or pressing ‘5’ on your keyboard will take you to the spatial performance tool. There are two types of spatial sensors: proximity triggers and zone triggers. By using these group triggers and also the SCENE GO button can be activated spatially.

Proximity sensors are circular and have a designated area with an epicentre. The closer the tracker gets to the centre the louder the sound or the higher the velocity. The strength of the relationship measured as a value between 0. and 1. whilst the trigger is activated. Moving within the range triggers the corresponding pad and mixes the fader up the closer the pointer gets to the epicentre of the event.

The zone sensors are based upon measuring the amount of activity from a live video stream within a designated area. Once a user adjusted activity threshold is
passed the trigger is activated. There is also a zone sensor specifically set aside for triggering the current cue and moving to another cue set by the user. This provides a means to dynamically remap your performance based on the movements or actions of the performers.

The GROUP triggers can be turned on or off with the button corresponding to their name. Likewise with the CUE trigger. The main difference between the two is that the cue trigger can only be a zone. The GROUP triggers can be toggled between a zone and proximity sensor. For a proximity sensor the area can then be set with the fader and then, by clicking move and then clicking again inside the performance area, assigned to the space. A zone sensor is has a sensitivity threshold to set and then by clicking move and dragging inside the performance area the zone can be assigned.

Figure 20 – The Space screen provides a novel way to perform sound design concepts with the area of performance and receive feedback between the software and participants
Zone sensors need a video feed to be running to activate them. The video will turn on automatically and appear in the background of the spatial screen once a zone sensor has been switched on. The video settings can be configured using the VIDEO SETTINGS button on the right hand side. The background can then be subtracted from the feed to increase the accuracy of the activity detector.

A background picture can also be added to describe a literal or abstract relationship to the space you are performing in. Clicking the “LOAD PICTURE” button will provide a dialog to browse for a .jpg or .png file. The “CLEAR” button will clear the background picture and all current triggers and their proximity/zone settings.

The spatial performance screen provides a way of dynamically controlling events in space and time. It could be used as a simple cross-fader, a multi-channel surround mixer or an interactive performance space. The x and y coordinates for mixing can either be controlled by left clicking and moving the mouse in the space or via MIDI.

In the case of the Space performance screen we can see demonstrated a literal and metaphorical extrapolation of narrative. It is an improvisational tool that engages with both performance action and design. Narrative is both created and reacted to in a visual manifestation of technical and creative relationships. There is a direct relationship to user interaction, performance and narrative.

Whilst exploratory interaction with the Space tool by moving the trigger point through the X/Y axis does reveal part of the nature of the tool it does pose issues in terms of affordance. Likewise with the zone motion based triggers. The nature and use of the tool may not be immediately obvious to a first time user. The space tool needs explanation and exploration before the user can discover the full potential of this performance screen. This will need further development and refinement as a conceptual tool.

**Group performance triggers**

Group triggers provide the user with a method for creating subsets of sounds to be launched simultaneously.
Four group triggers can be set arbitrarily on a per scene basis. They can be preset or set on the fly during a performance. A user sets the group by holding down the ‘g’ key on the computer’s keyboard and cycling through the group states from unassigned, to group 1, then group 2, group 3, group 4 then back to the unassigned state. Below is an example of triggers assigned with group relationships:

![Image](image.png)

**Figure 21** - Groups are assigned with a keyboard shortcut to correspond to one of four group trigger buttons on the left-hand side.

Once set, the corresponding colour of the group will highlight the outside of the trigger button. The triggers still function, as they would do normally. They can be pressed to trigger their individual sound event. The group triggers are numbered and coloured to correspond with the trigger button states. When the group trigger they are assigned to is pressed all triggers in the group will be activated in sync. This is a way of leveraging the performance capability of a scene.

The group trigger buttons, which are located on the left hand side of the interface, automatically resize to become larger when the mouse is over them. This can be seen in Figure 22. The dynamic resizing of an interface element in this case can
address miss-hit issues during a performance. It is an idea that is only utilized for the moment in the group triggers but could be employed in other areas of the interface that are performance critical.

Figure 22 - Dynamic resizing of group trigger buttons minimizes mishits during use
The overview window

Figure 23 - The overview menu window provides realtime feedback of the state of the software during a performance.

The overview window has been designed to be either open or closed during a performance. It contains status and monitoring data that could be useful to users but is not necessary in terms of core performance information.

At the top is a scene menu. Users can name scenes so that they can be identified while navigating with the main transport. These scenes can have notes or an
accompanying description of the makeup of the scene. New scenes can be appended to the end of the list or inserted in between two other scenes. Scenes can also be renamed, moved and deleted. This allows the flexibility needed to respond to changes in narrative structure.

Output delays can be used to adjust the timing of output channels where speakers are separated. This can also be used to affect a delayed sound for aesthetic on narrative purposes.

The fader overview is an interactive status display of the state of all twenty four faders. A ‘panic’ button is included in the top right hand corner to set all faders to zero output.

Below that individual or combinations of input and output channels can be monitored via VU (volume units) meters or through a sonogram display.

**The MIDI Learn screen**

Access to the MIDI Learn configuration screen is through a button at the bottom of the Main performance window. It allows certain parameters to be controlled via external input from devices outputting MIDI note and controller messages.

![The MIDI Learn window](image)

*Figure 24 - The MIDI Learn window extends the affordances of the software by allowing external devices to control it’s functions*
The trigger edit windows

Holding down the ‘e’ key on the keyboard and clicking on the desired trigger give the user access to the edit window for each trigger pad. This window is important, as it is the means by which the user defines the audio elements for a scene.

There are three main functions that each trigger pad can perform. It can play back sound files, mix and output incoming sound through available input channels and output MIDI note and controller change messages. Only one mode of operation is viable per trigger for a given scene. There are also two main modes of operation: manual triggering and cued. The manual trigger mode requires user input to activate its function. The Cued mode will activate the trigger when the scene is set. In cued mode the trigger can also be manually retriggered at any point.

If a tempo is set in the top left hand corner the loaded sample, input or MIDI data can be retriggered at a set interval measured in beats per minute (BPM). If multiple triggers are set at differing intervals the trigger grid functions like a drum
machine or sampler/sequencer. Sequencing and looping can afford an organic manipulation of the sonic narrative.

The input function also provides a novel method of interaction. A threshold level can be set and when a sound from received through that input exceeds that level a trigger can be sent to activate the cue the next scene of trigger another pad.

Output channel routing specific to the sounds for this pad are also set here. This does not apply however to the MIDI output. Fader or velocity levels, depending on the function set for the pad, can be automated when the pad is triggered. The graphical script can be used to describe complex fades or changes to velocity over time.

A single VST (Virtual Sound Technology) plugin can be loaded per trigger as well. VST effects can extend the scope for altering the quality of sounds being played. Thousands of diverse effects, both commercial and free, are available to be explored by users.
The program menu

The program menu is located on the right hand side of the main screen so as to minimize its impact on user focus. It provides a variety of functions to the user.

At the top is the record button. When pressed the user is prompted via a dialog to save a recording of the performance. The program will then wait for a signal to start recording. This is a useful tool because it provides a reference to the user and collaborators. A recording of the sound output could be used to review and hone performance skills and explore narrative successes and failures within the context of the time they occurred.

Underneath are options to set and monitor the sound driver being used by the software. Below the driver options are the buttons to open the Overview window and the Audio Setup window.

The show section allows the user to create, load and save shows. Finally the Quit button prompts the user with a dialog to exit the program.

Figure 26 - The program menu provides links to the utilitarian functions of the software
3.3 Constructing a sound design

The interaction model provided by the software allows for the programming of a sound design. Changes in the design can be added in steps without interrupting the flow of the sound itself. Communication can come from directors, performers, participants, designers, and user self assessment and be modified on the fly to be incorporated immediately back into the performance environment.

The interface also affords the user the ability to create narrative palettes or sound boards for scenes to try different moods, tones, tempos and timbres live with the performance and quickly move on or reassemble elements that are or are not working.

3.4 The software explained in terms of user interface principles

Detailed in the table below is a basic summation UI principles. I will now elaborate and the way these principle informed and where utilized through the design of the soundNode interface.

<table>
<thead>
<tr>
<th>PRINCIPLE</th>
<th>CONCEPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The principle of user profiling</td>
<td>Know who your user is.</td>
</tr>
<tr>
<td>2. The principle of metaphor</td>
<td>Borrow behaviours from systems familiar to your users.</td>
</tr>
<tr>
<td>3. The principle of feature exposure</td>
<td>Let the user see clearly what functions are available</td>
</tr>
<tr>
<td>4. The principle of coherence</td>
<td>The behaviour of the program should be internally and externally consistent</td>
</tr>
<tr>
<td>5. The principle of state visualization</td>
<td>Changes in behaviour should be reflected in the appearance of the program</td>
</tr>
<tr>
<td>6. The principle of shortcuts</td>
<td>Provide both concrete and abstract ways of getting a task done</td>
</tr>
<tr>
<td>7. The principle of focus</td>
<td>Some aspects of the UI attract attention more than others do</td>
</tr>
</tbody>
</table>
The main user persona this software is aimed at is sound designers. Defining a sound designer in itself is a complex task. Sound designers can have varying levels of expertise, experience and backgrounds. Other users throughout the environment must also be considered. The impact on directors, performers, stage managers, artists, musicians and audience members must all be taken into account in a system such as this.

The focus of the software architecture is primarily on the construction and performance of a sound narrative though. The design therefore affords all its attention to the elegant and economical interaction between user, performer and text.

The soundNode systems performance screens employ visual metaphors that should be familiar to any user of audio software. The pads are modelled from software and hardware controllers such as drum machines, MIDI controllers and keypads. The faders follow the basic vertical taxonomy of a mixing desk. The filter screens are modelled on parametric equaliser units, found in audio software and hardware everywhere. The dials of the speed screen are also like pots on a mixing desk.

Although there are five performance screens the information is kept within the one window. Tabs and hot keys keep the flow of interaction relevant and focused to
this area. All other windows that are unnecessary to the performing of the design are hidden from the user.

Both the design of the graphical user interface (GUI) and programming are consistent throughout the software’s architecture. Colour palettes, style sheets and prototype objects maintain the consistency of the program visual. The programming code is object oriented and follows a hierarchical taxonomy of flow from a core controller down the more specific functions.

Visual feedback is given for all major user actions. Buttons have mouse-over and click states, dialog and hint boxes are employed to guide interaction and automated features update in real time. Where possible a variety of methods can be employed by the user to complete the same task. The focus of this is in the trigger of events for playback. The Space performance screen for example provides a more abstract method of trigger events than just clicking the buttons on the Trigger screen.

The focus of the users attention has been directed to the largest part of the main window at all times, the performance screens, through the use of scale and visual flow. The language of the UI is generated through established visual and metaphorical paradigms in the fields of audio software, sound design and HCI. Where possible there is as little deviation from the principles of the reference design metaphors.

Guidance to users is provided through configurable mouse over hints, dialog boxes to highlight the nature of decisions, strong visual feedback from user input and software output and a clear taxonomy of information in the design.

During the creation and performance of sound designs there are several ways that users can revert or censor their actions or mistakes. The ability to save shows and keep several copies of variations of a show provides one of the strongest ways methods of protection for users. On a task-by-task basis users are able to stop the playback or single sounds or the entire scenes playback. They can also unload wrongly placed files; delete, modify, move and copy scenes; revert settings on performance screens to defaults and manually adjust for errors during performance.
Major decisions such as quitting the program, saving over show files and loading new shows are buffered by user dialog prompts.

Only a single performance screen can be accessed at a time. Users can clearly delineate their choice of action and economise their input. Although the aesthetic of the program is a mostly subjective and variable factor between all potential user types every effort has been made to provide a clean and clear visual design.

3.3 The software’s engagement with narrative

There are constraints that arise from the mechanics of interaction with the software interface that must be acknowledged. Input via the keyboard and mouse can be counter-intuitive and deflect attention from the environment and agents of the performance. These are issues that can be circumvented though through practice and familiarisation of the user with the structure of the interface and by using alternate input methods, such as MIDI or USB control devices suited to the experience and application desired by the user.

For instance while user input via the mouse is one method of control for the Space tool external MIDI controller values can be used to control its X and Y coordinates. Motion tracking software could even be interpreted to provide MIDI controller change values to provide an immersive experience to user interaction.

A further development to this tool in the future could extend its immersive narrative experience. Scenes could be dropped in as trigger points. This could become a hypertextual experience for the performer and computer agent. A choose your own adventure defined by manual interaction from sound performer and communication with the performer/s.
Chapter 4 - Conclusion

4.1 Conclusions

"Nodal plots are a series of noninteractive events, interrupted by points of interactivity. This plot structure, which provides the most potential support for the classic dramatic arc, has been referred to as a "string of pearls." [...] Nodal plots have a single beginning and usually at least two endings. It’s important to note that while the event of the ending will be the same it does not need to happen at the same time in the story. In some cases there is a goal at the end of the story. This is generally the case with games. “(Meadows, 2002)

As technology in the performing arts has evolved and progressed there has been a struggle for performers, participants and observers to reconcile, integrate and find relevancy in the emergent methodologies they represent in their practices. The interfaces presented in these technologies are affording the user agents are broader array of scope for interaction and a dialogue with performance texts, especially in the sense duplex feedback on information and real-time processing. Software and hardware interface designers are still however negotiating the effective production of powerful and elegant means of communicating these concepts.

The name of the software I have constructed – soundNode implies that it is interconnected in a system, an environment that offers many pathways. As Stephen Matthews suggests in his definition of Nodal plots this point of connection is a system of decisions that are governed by the rules and conditions set by the design of the narrative structure. Each node is a nodal system in itself that follows the rules determined by the architecture of the system that it resides within but from which user agency can determine the flow to the next node and the decisions made within that node.

The soundNode software model aims to provide a system where users can work with performers, within the constraints of the narrative, to provide an interactive experience that is both useful and exciting. It is a system that users, performers and observers must be willing to embrace to make it effective and it certainly would not work in every narrative context. It is a model that will keep developing
though as a method for bridging the gap between linear, reactive systems and free form gestural and immersive systems.

4.2 Future directions

Many refinements and bugs still need to be addressed in the system. To this end an upgrade path and list of known issues has been included in appendices A and B respectively.

More extensive user testing must be undertaken and evaluated to produce the interface refinements. This is a process I hope to address by releasing a beta evaluation version for public feedback. The software will also be released as an open source project to the Cycling '74 community so that the ideas here can be extrapolated and experimented with.
Bibliography


Index

A

A Preliminary Poetics ........................................... 25
Ableton Live .................................................... 29, 31, 32
Alan Dix .............................................................. 27
Alice Bayliss ...................................................... 27
Apple iPhone ...................................................... 37

B

BPM

Beats Per Minute .................................................. 57, 72
Brenda Laurel ....................................................... 14, 20, 21, 24, 26
Brian Chipperfield .................................................. 15

C

channel .................................................................. 52
Charlie Richmond .................................................. 28, 30
Cognitive Load Theory .......................................... 15
Computers as Theatre ............................................ 14, 23, 24, 26
Creative Commons ............................................... 4
Cricket v2 ................................................................ 46
cross-fader ............................................................ 52
Cycling ‘74 .............................................................. 2, 4, 35

D

DAW

Digital Audio Workstation ........................................ 72
Donald Norman ..................................................... 14, 22

H

Hamlet On The Holodeck ........................................... 23
HCI

Human Computer Interaction .................................... 72
Human Computer Interaction .................................... 11, 14

I

IxD

Interaction Design .................................................... 72

J

Jakob Nielson ......................................................... 14
James J. Gibson ..................................................... 22
Janet Murray ......................................................... 14, 23
Jared Dunne .......................................................... 33
Jef Raskin .............................................................. 14
Jennifer G. Sheridan .............................................. 27

K

keyboard ................................................................. 50

M

Mark Stephen Meadows .......................................... 18
Max/MSP/Jitter ....................................................... 2, 4, 11, 35
Michael Mateus ..................................................... 14, 23, 25
MIDI ................................................................. 20, 28, 30, 35, 40, 42, 46, 56, 57, 58, 61, 63
Musical Instrument Digital Interface .......................... 44, 72
Monome ............................................................... 29, 33, 34, 41
Neo-Aristotelian model ........................................... 2, 34
Nintendo Wii .......................................................... 29, 32

O

Open Sound Control ................................................ 37

P

Performance Triad Model ........................................ 27

Q

QLab ................................................................. 29, 30, 46

R

Richmond Sound Design .......................................... 28
Rolf Molich .......................................................... 14, 15

S

Simon Lock ............................................................ 27
Software Requirements Specification
SRS ........................................... 43
spatial ....................................... 50
Stage Research’s SFX .................... 46
Stephen Matthews ....................... 64
Steve Dixon ................................ 19
SysEx ....................................... 30

U

UI 35, 60, 62

User interface ................................. 72
UXD
User Experience Design .................. 72

V

VST ........................................... 58, 69
VURP
(Viewer, User, Reader, Player) ........... 72
Appendices

A. Upgrade path

Below is a list of desired or as yet undeveloped features that may be included in future versions of the software.

- OSC support
- Global tempo network/MTC synchronisation
- Extensive reworking of VST support
- Customization of interface (colour schemes)
- Screen resolution support
- Automation for Filter, Speed and Space tools
- Follow-on cues can jump to any scene (partial implementation in zone and proximity sensor triggers)
B. Known issues

Below is a table of all known issues outstanding with the software included with the thesis, at the time of the submission of this research.

<table>
<thead>
<tr>
<th>BUG/ISSUE</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pause/Play state of individual triggers can occasionally fall out of sync with global Pause/Play control</td>
<td>Can effect performance and audio playback. Sometimes play/pause button must be hit twice to start/stop sound.</td>
</tr>
<tr>
<td>CPU performance issue when all triggers activated</td>
<td>When samples are loaded into all twenty-four triggers CPU usage on the test system hovered at around 45-50%. This load may be excessive or unacceptable for some systems.</td>
</tr>
<tr>
<td>Cue/show save issues</td>
<td>Cues or whole shows may not properly save if a show is not loaded or created from the outset. This needs further exploration.</td>
</tr>
<tr>
<td>Issues in layering of video and images in the Space tool.</td>
<td>Currently overlay images must be loaded after the video is instantiated for it to not be overwritten on screen.</td>
</tr>
</tbody>
</table>
C. Recommended system requirements

**soundNode (2011):**

**Macintosh:**

Any G4 or faster (Intel Mac, PPC unsupported)

1 GB RAM (2 GB of higher recommended)

Mac OS X 10.4.11 or higher

**PC:**

1.5 GHz CPU or faster

1 GB RAM (2 GB or higher recommended)

Windows XP, Windows Vista, Windows 7

Windows-compatible soundcard (preferably with an ASIO driver)

**Linux:**

Varying success with Linux systems using WINE, no recommended system. As yet unsupported.
D. Glossary

• BPM – Beats Per Minute
• DAW – Digital Audio Workstation
• HCI – Human Computer Interface
• IxD – Interaction Design
• MIDI – Musical Instrument Digital Interface
• MIDI SysEx – Musical Instrument Digital Interface System Exclusive
• UI – User Interface
• UXD – User Experience Design
• VURP – (Viewer, User, Reader, Player)
Author/s:
Stewart, Robert Andrew

Title:
soundNode: a user interface for sound design in performance environments

Date:
2010

Citation:

Persistent Link:
http://hdl.handle.net/11343/36409

File Description:
Final thesis

Terms and Conditions:
Terms and Conditions: Copyright in works deposited in Minerva Access is retained by the copyright owner. The work may not be altered without permission from the copyright owner. Readers may only download, print and save electronic copies of whole works for their own personal non-commercial use. Any use that exceeds these limits requires permission from the copyright owner. Attribution is essential when quoting or paraphrasing from these works.