Musical Instrument Design Process for Mobile Technology

Timothy J. Barraclough
Victoria University of Wellington
Wellington, New Zealand
barractimo@myvuw.ac.nz

Dale A. Carnegie
Victoria University of Wellington
Wellington, New Zealand
dale.carnegie@vuw.ac.nz

Ajay Kapur
California Institute of Arts
24700 McBean Parkway
Valencia, California
akapur@calarts.edu

ABSTRACT

This paper presents the iterative design process based upon multiple rounds of user studies that guided the the design of a novel social music application, Pyxis Minor. The application was designed based on the concept of democratising electronic music creation and performance. This required the development to be based upon user studies to inform and drive the development process in order to create a novel musical interface that can be enjoyed by users of any prior musicianship training.

Author Keywords

Iterative design, network music performance, music app, novel sequencer, interface design, user testing

ACM Classification

D.2.2 [Software Engineering] Design Tools and Techniques H.5.5 [Information Interfaces and Presentation] Sound and Music Computing, H.5.2 [Information Interfaces and Presentation] User Interfaces

1. INTRODUCTION

When designing a novel musical interface the hardest task is creating a user experience that is intuitive to people of varying skill level. To create an interface that is simple for beginners to grasp may result in a lack of musical depth or expressivity. Conversely, an interface that provides a high level of musical depth and expressivity often will result in a more complex interface that may alienate beginner musicians. This issue can be mitigated by using a design process based on feedback from users with a range of prior training to discover the requirements of a whole spectrum of potential users.

This paper presents Pyxis Minor, a new social music application for iOS devices and its iterative design process. Integral to the iterative design were user studies that gathered feedback to guide the application development.

Hundreds of musical instrument applications exist in Apple's App Store, however the functionality of these applications incorporates features that target a specific user base. For example, KORG's WIST technology [5] allows for applications to synchronise metronomes between devices. It is only available for applications that require prior musical experience. Most of these applications use musical interaction metaphors based on digital audio workstations (i.e. KORG Gadget[1]), traditional keyboard synthesizers (i.e. Arturia iSEM[2]) or traditional rhythm sequencers (i.e. Fingerlab's DM1[3]). This indicates that the ability to synchronise between devices is only relevant to experienced musicians. This has not been fully explored for beginner-friendly interfaces. The tendency to provide more musically deep or expressive features for interfaces furthers the conceptual schism between beginner musicians and those with prior training.

On the other hand, mobile based musical instruments that target beginner musicians have demonstrated wide commercial success, for example the Smule applications Ocarina [9] and Magic Fiddle [11]. Fels and Blaine state "providing novices with easily accessible music making experiences is more important than having a complex interface with built-in, upward capability for virtuosic expression"[2], but ideally we can create a meaningful experience for more experienced musicians by including upward capability that does not require a complex interface. Applications targeted towards beginner musicians can provide a novel experience, but may lack expressive musical depth.

By beginning with a simple and intuitive interface that appeals to beginner musicians, we can add features to increase the musical depth and expressivity to appeal to experienced musicians; this can open discussion about the democratisation of electronic music processes. Tanaka explains this democratisation process in regards to NIME practices, "the explosion of consumer devices with built-in sensors such as mobile phones and game controllers such as the Nintendo Wii- remote have opened up and democratized NIME practice outside the realm of academic research" [8] but we argue that, by extension, through iterative design methodology the democratisation can extend to wider musical practices (specifically that of electronic music process).

2. METHODOLOGY

Iterative design methodology is key for application development and Neilson suggests that the biggest gains to the usability of a interface will occur in the first few design iterations [7]. Our intention was to design an application for a diverse user base, so understanding the requirements the user base is integral to the success of the application. By improving, iterating and testing the application with a representation of the user base we can create an application that succeeds at achieving the goal of the application: to help the democratisation of electronic music processes, al-

[1]https://itunes.apple.com/app/id791077159
[2]https://itunes.apple.com/app/id673921187
[3]https://itunes.apple.com/app/id431573951

NIME'15, May 31-June 3, 2015, Louisiana State Univ., Baton Rouge, LA.

lowing users an enjoyable experience of creating electronic music, regardless of prior musical experience. Additionally, the prioritisation of feature development can be proportionally distributed as per issues raised from user feedback. This iterative design process echoes the core artistic intention of the project - democratisation of electronic music creation.

3. ITERATIVE DESIGN

The concept for the application is to have a visual grid where users place nodes which represent musical notes. The co-ordinates of the nodes represent pitch-velocity pairs from the x and y axis, respectively. Users input a sequence of nodes to make a musical loop, which is recorded in real-time. The nodes of the sequence are quantised to a chosen tempo. This allows a user to make music with a strong metric pulse. Users can create up to four different sequences (identified with symbols/colours) that run simultaneously.

A goal of the design is to show users musical concepts and the application functionality, rather than to explicitly give direct instruction. This was intended to allow users playful exploration in order to understand functionality. This philosophy is intended to help democratise electronic music processes by revealing that the creation of electronic music is, in fact, based upon exploration. We adhered to minimalist aesthetics to create a simple, intuitive interface. A specific focus was placed upon simple interfacing that reveals concepts integral to understanding electronic music processes to users.

3.1 First Iteration

The first iteration of the application was prototyped for Mac OSX. It was built using openFrameworks [6] for the user interface and ChucK [10] for logic and sound design. Users interacted using a trackpad and keyboard. This iteration functions as a proof of concept, developing the game-type logic.

An iOS application was built in order to control a separate cursor in the Mac OS X using the Open Sound Control (OSC) protocol over a UDP network. The iOS application used data from the accelerometer, filtered through a custom conformal mapping algorithm, to control the cursor's position on the Mac application.

3.1.1 Results

Navigating the grid using the accelerometer to control a cursor pointer with the accuracy required for fine musical control was more difficult than anticipated. The novelty of this control scheme does not outweigh the difficulty. Additionally, having the interaction split between two devices makes it difficult for users to understand where to focus their attention. It is more intuitive to have the user interact on a computer. The conformal mapping technique has been also tested as a controller scheme for audio effect parameters (i.e. manipulating a filter cutoff and resonance simultaneously with the x and y values). This appears to be a better use of the accelerometer data.

3.2 Second Iteration

The second iteration required only one device. The application was rebuilt for iPad, iPod Touch and Mac OS X. Usability tests were conducted to deduce which method of interaction and which hardware interface was most suitable. This iteration used the Cocos2D [4] game engine and libPD [3] audio engine for creating a cross platform application. Aside from the method of interaction, the three hardware implementations were fundamentally identical. The iPad and iPod Touch both used screen touches for interaction, whereas the OS X implementation used a mouse. A screenshot of the OS X version can be seen in figure 1.

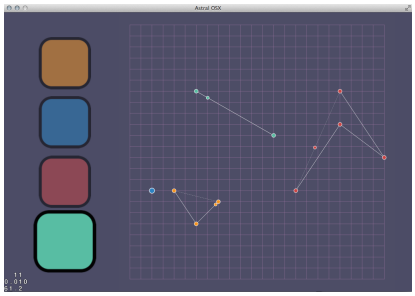

Figure 1: OS X version of the second iteration.

3.2.1 Pilot Study

A pilot study was conducted with five users of different musical backgrounds and levels of formal training in order to gauge which hardware interface-interaction pair was the most suitable. The most significant finding from this study was that the interaction on the iPod was not sufficient. The results show the most prominent problems for the iPod device were the "Ease of achieving the full range of available interaction" and the "Simplicity of the Interaction". User quotes suggest this is due to the smaller screen of the iPod touch. One user indicated that despite the iPod being more difficult to use, the compact nature of the hardware is an element that the other contexts do not share and could be leveraged in a different version. While the Mac and the iPad remained stationary on the table, the iPod was often picked up and mobilized. When asked about new features, four out of five users responded they would like further options to control parameters of the sound produced. Suggestions on how to do this focused predominantly on DSP effects and sound generator possibilities. Additional usability issues were revealed which were corrected in the next iteration of development.

3.3 Third Iteration

The third iteration of the application design has focused on fixing bugs and generally improving the responsiveness, speed and usability of the application. This required a complete rewrite of the audio engine to both decrease the processing power required and to allow users to change various basic audio parameters. The audio engine was written using Apple's Core Audio Framework[1] Additionally, more options were added to avoid the aforementioned usability issues. The visual aesthetic was also greatly improved.

3.3.1 User Study

This iteration was set up as a demonstration at the Sonic Engineering Expo 2014 at Victoria University. A photo of this can be seen in figure 2. Users were given a brief demonstration and shown the short tutorial before being asked to spend approximately 10 minutes using the application on all three devices before fulfilling a questionnaire.

Both the iPad and iPod versions performed better than the previous iteration but users did not respond as well to the changes made to the OS X user interface. The iPod Touch, overall, only did marginally better than the Mac computer. This could be due to the increased amount of navigation menus to alter sound parameters and the added functionality for manipulating the audio sequences. Although equally intuitive, the navigation with a cursor was

[4]http://www.cocos2d-swift.org/

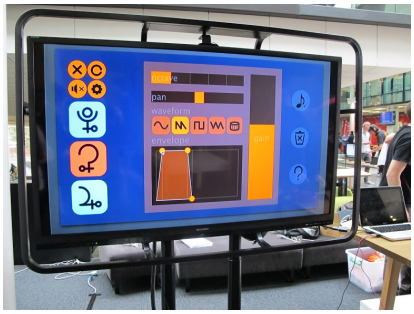

Figure 2: OS X iteration of the third iteration as displayed at the Sonic Engineering Expo 2014.

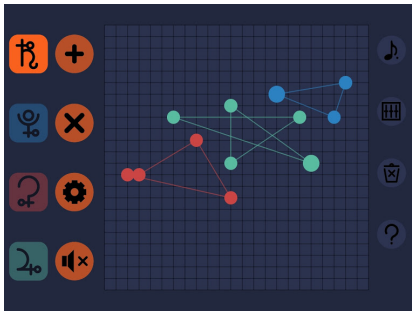

Figure 3: Screenshot of the first published version of Pyxis Minor on iPad.

substantially slower than with the touch devices. As such, the improvements made to this iteration indicated that the application design (divorced from the hardware contexts) improved the user experience.

Users were asked their preferred platform. The iPad was the winner, followed by the iPod Touch, possibly due to the relative portability of the devices. Although the iPod Touch was the smallest and most portable device, the iPad appeared to be the best balance of portability and comfortability. Users cited that interacting with some of the UI elements was difficult on the iPod Touch.

3.3.2 Analysis

The user study results and informal feedback revealed direction for the next iteration of development:

- Users requested more audio manipulation options (such as effects).
- The additional controls for the sequences were difficult to access due to the small size of the buttons.
- Multi-touch was requested for the iOS devices to afford more complex harmonies.
- A more comprehensive tutorial was requested.

The results from the platform usage indicate that despite improvement for the application on iPod Touch, there is still room to leverage the handheld nature of the device. Interestingly, the ability to change the musical scale or key had not yet been requested by any user. This suggests that the issues and requested features up until the next iteration are more important than something arguably integral to musical expressivity.

3.4 Pyxis Minor

The application was named Pyxis Minor and published to the Apple App Store for iOS. The first app store version is an upgrade to the third iteration, including optimisation for iOS devices. A screenshot of the iOS version can be seen in figure 3. UI elements were repositioned and resized to allow for easier use on smaller iOS devices. Multi-touch was added to allow another degree of musical expressivity.

3.4.1 Results

Although distributed as a universal iOS application, the amount of downloads for iPad far outweigh the amount of downloads for iPhones and iPod Touch, with iPad accounting for 65% of all downloads (currently 1697). Most downloads come from users browsing the music section of the app store. This indicates a tendency for the user base to be more musically inclined people, however this could also be due to the App Store classification system. Multiple reviews and comments on various internet forums requesting functionality to connect to other iOS applications via AudioBus, Inter-App Audio or Core MIDI indicate that users are more musically inclined.

3.5 Audio Effect Update

Aside from minor bug fixes and tweaks, the biggest update to Pyxis Minor in version 1.1 is the addition of audio effects to the signal chain. We incorporated audio effects from the Synthesis Tool Kit (STK) [4]. These additional audio effects can be accessed from the main application screen (seen in figure 4). In this screen users are able to select from six kinds of effect and control two parameters of the effect simultaneously with an x-y pad and the wet/dry amount with an additional fader.

An additional option to enable a metronome was added. This was in response to some users expressing difficulty in understanding how their loops were being quantised. The metronome features a high hat pattern fitting with the sonic aesthetic of the application.

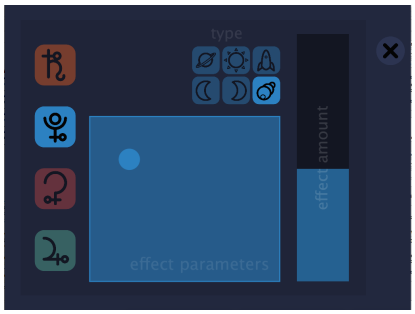

Figure 4: Effects menu screen (pictured on iPad).

3.6 Tutorial Update

A more extensive tutorial menu was added to version 1.2. This provided a reference and brief introduction to the applications core functionality. Informal feedback has indicated that users gain more satisfaction from the application when they are comfortable with the interaction and have realised that it posits no task or goal orientation. Users are encouraged to find their own meaning and guide their own interaction as they see fit, similarly to how an adept musician would approach a musical instrument.

The choice to have a more in-depth tutorial reflected a difficult decision in the development cycle. The application was built with the idea of it being intuitive, with a learning process focused on allowing the user to discover functionality without explicit direction. This became problematic as the complexity of the application advanced when more

features were added. Some of the functionality uses musical jargon and so a more comprehensive tutorial would helps ensure people without prior musical experience do not feel alienated. Additionally, due to the lack of explicit goal orientation for the user, some users expressed confusion over what they are supposed to do with the application. This confusion is resolved with more explicit direction as to the mechanics of the application.

3.7 Social Collaboration

Until this iteration, Pyxis Minor had focused on the individual process of creating electronic music, yet neglected the experience of playing music with other people, which is arguably one of the most rewarding elements of playing music. We hypothesised that an additional layer of musical complexity could be added by the inclusion of a social element. The application was further developed to allow 2 users to sync their metronomes in order to collaboratively make music together using Korg's WIST framework[5].

3.7.1 User Study

A user study was conducted with pairs of participants. Users were asked to spend time familiarising themselves with the iPad version of the application and to explore solo play. Users were then able to play as a duo before filling out a short questionnaire. The investigator was present to help with any questions, although encouraged the users to help each other where possible. The results strongly indicated that the networked metronome improved the user experience. All the surveyed users indicated that the music resulting from paired interaction sounded better than their solo interaction.

Users also indicated that the inclusion of the comprehensive tutorial aided in their understanding of the core mechanics of the application from the outset. This improved the user experience overall - suggesting the tutorial update achieved its specific goal.

3.7.2 Analysis

Out of a possible 40 points, users gave 34 points indicating the degree to which the collaborative aspect created a social experience. This, in combination with the fact that only approximately half the users indicating that they were verbally discussing techniques or musical direction suggests the social element of the application is in the shared musical experience between the participants. This is justified by users giving 75% of the maximum possible score when asked whether they felt they were making music together". This shared networked functionality, although normally reserved for the domain of musical apps targeted to more experienced musicians, dramatically improved the user experience and added musical complexity without adding unnecessary interface complexity. The overall results from this user study suggest that the process of creating a simple musical interface/instrument and then augmenting the user experience through iterative design process has led to an application that minimises application complexity, but still provides a level of musical complexity and expressivity suitable for musicians of varied prior musical backgrounds.

3.8 Future Work

Pyxis Minor has progressed from the original concept to published application. Future development will be directed by user feedback. Some users have expressed difficulty with understanding and manipulating the timing mechanism. Adding a traditional step sequencer view will allow users to both view and edit their sequences with respect to the time domain.

An additional layer of effects will allow more control of the sound. This layer will be controllable using the accelerometer control. This feature will likely be of prominent when users are collaborating together.

The user interface for the networking functionality still has a few bugs. Once these have been addressed, the publicly available version will be updated to include the networking functionality.

4. CONCLUSION

The iterative design process that lead to the creation of Pyxis Minor has resulted in an application that can begin a dialogue about the democratisation of electronic music processes. By informing the development upon user feedback the resulting application can be of interest to both musicians of any training. The implemented features represent a middle ground between the expectations of multiple users with varied prior musicianship training. By structuring the development of the application iteratively, Pyxis Minor has proven that we can create applications that not only allow users of opposite ends of the spectrum a tailored musical experience, but can create applications that in themselves are a work of art. The design philosophy, which progressively introduces features, allows Pyxis Minor to continue to meet the requirements of users and evolve in order to help lower barriers to entry to electronic music creation and give new musicians an opportunity to create music, socially, whilst remaining musically interesting.

5. REFERENCES

[1] Apple Inc. Core audio overview, 2011.

[2] T. Blaine and S. Fels. Contexts of collaborative musical experiences. In *Proceedings of the 2003 conference on New interfaces for musical expression*, pages 129–134. National University of Singapore, 2003.

[3] P. Brinkmann, P. Kirn, R. Lawler, C. McCormick, M. Roth, and H.-C. Steiner. Embedding pure data with libpd. In *Proceedings of the Pure Data Convention*, 2011.

[4] P. R. Cook and G. P. Scavone. The synthesis ToolKit in c++ (STK), 2014.

[5] Korg Inc. KORG WIST | wireless sync-start technology, 2014.

[6] Lieberman, Zach, Castro, Arturio, Watson, Theo, and others. openFrameworks.

[7] J. Nielsen. Iterative user-interface design. *Computer*, 26(11):32–41, Nov. 1993.

[8] A. Tanaka and others. Mapping out instruments, affordances, and mobiles. In *Proceedings of the International Conference on New Interfaces for Musical Expression. Sydney*. NIME, 2010.

[9] G. Wang. Designing smule's iphone ocarina. In *Proceedings of the International Conference on New Interfaces for Musical Expression. Pittsburgh*, volume 291, 2009.

[10] G. Wang, P. R. Cook, and others. ChucK: A concurrent, on-the-fly audio programming language. In *Proceedings of the International Computer Music Conference*, pages 219–226. Singapore: International Computer Music Association (ICMA), 2003.

[11] G. Wang, J. Oh, and T. Lieber. Designing for the ipad: Magic fiddle. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, volume 30, 2011.