Prototyping of Ubiquitous Music Ecosystems

Victor Lazzarini[1], Damián Keller[2], Carlos Kuhn[3], Marcelo Pimenta[3], Joseph Timoney[1]

[1]Sound and Music Research Group
National University of Ireland, Maynooth
Co. Kildare Ireland

[2]Amazon Center for Music Research - NAP
Universidade Federal do Acre - Federal University of Acre

[3]Computer Science Department
Universidade Federal do Rio Grande do Sul

{victor.lazzarini,joseph.timoney}@nuim.ie,

dkeller@ccrma.Stanford.EDU, {mpimenta,ckuhn}@inf.ufrgs.br

Abstract. *This paper focuses the prototyping stage of the design cycle of ubiquitous music (ubimus) ecosystems. We present three case studies of prototype deployments for creative musical activities. The first case exemplifies a ubimus system for synchronous musical interaction using a hybrid Java-JavaScript development platform, mow3s-ecolab. The second case study makes use of the HTML5 Web Audio library to implement a loop-based sequencer. The third prototype - an HTML-controlled sine-wave oscillator - provides an example of using the Chromium open-source sand-boxing technology Portable Native Client (PNaCl) platform for audio programming on the web. This new approach involved porting the Csound language and audio engine to the PNaCl web technology. The Csound PNaCl environment provides programming tools for ubiquitous audio applications that go beyond the HTML5 Web Audio framework. The limitations and advantages of the three approaches proposed - the hybrid Java/-JavaScript environment, the HTML5 audio library and the Csound PNaCl infrastructure - are discussed in the context of rapid prototyping of ubimus ecosystems.*

1. Introduction

Creativity-centered design of ubiquitous musical systems involves at least four developmental stages: defining strategies, planning, prototyping and assessment. This paper focuses on the third stage of the design cycle, prototyping. The first section shows related works in the field and the second places the activity of prototyping within the context of ubimus design. Then we present a case study focusing on the deployment of a ubimus system for synchronous musical interaction using a hybrid Java-JavaScript development platform based on browser technology. The second case involves the use of Web Audio in HTML5 to implement a loop-based sequencer. And the third case features a simple example of an HTML-controlled sine-wave oscillator using the Csound PNaCl programming environment. The final section provides a summary of the observations gathered

during the design of these three prototypes and discusses the limitations and advantages of each approach.

2. Related work

In recent years, there has been some research (and commercial) work aiming to provide support for development of audio applications for mobile platforms like MobileSTK [Essl and Rohs 2006], based on STK and released in 2006, with support for Symbiam and Windows CE devices. This platform was also ported do iOS in 2010 [Bryan et al. 2010] and incorporated in a toolkit called MOMU. Also from Essl [Essl 2010], we have Urmus, a LUA framework that is a multi-layered environment intended to support interface design, interaction design, interactive music performance and live patching on multi-touch mobile devices. Control [Roberts 2011] is an application that allows users to define custom graphic interfaces for MIDI and OSC. The interfaces are defined using web standards like HTML,CSS and Javascript. Roberts also is one of the creators of Gibber [Roberts et al. 2013], a language for live-coding in the browser. Gibber also has a 2D drawing API and event handlers for touch, mouse, and keyboard events, enabling fast prototyping. Since Gibber is centralized on a server, users can create collaborative programming sessions and publish compositions and instruments.

3. Designing ubiquitous music systems

Defining design strategies for ubiquitous music encompasses two areas of expertise: interaction and signal processing. The Ubiquitous Music Group (g-ubimus) has been investigating the musical applications of methods based on human-computer interaction and ubiquitous computing techniques. Metaphors for interaction provide abstractions that encapsulate solutions applicable to a variety of activities without making unnecessary technical assumptions [Pimenta et al. 2012]. Thus, interaction metaphors materialize general ergonomic principles to fulfil the human and the technological demands of the activity [Keller et al. 2010] [Pimenta et al. 2012]. On a similar vein, recurring technological solutions can be grouped as interactions patterns [Flores et al. 2010]. These patterns are particularly useful when developers face the task of finding suitable strategies to deal with specific interface implementation issues. So far, our group's research has unveiled four musical interaction patterns: natural interaction, event sequencing, process control and mixing [Flores et al. 2012]. Each of these patterns tackles a specific interaction problem. Natural interaction deals with forms of musical interaction that are closely related to handling everyday objects. Event sequencing lets the user manipulate temporal information by freeing the musical events from their original time-line. Process control provides high-level abstractions of multiple parametric configurations, letting the user control complex processes by using simple actions. Mixing can be seen as the counterpart of event sequencing for synchronous interaction. Musical data - including control sequences and sound samples - is organized by user actions that occur in-time. Furthermore, technologically based musical environments also demand tailoring support for sound rendering. Signal processing techniques for creative musical activities have to be developed according to the characteristics of the tasks involved in the creative cycle, the computational resources provided by the support infrastructure and the profile of the target users. Ubiquitous musical activities may involve mobility, connectivity and coordination among heterogeneous devices with scarce computational resources. Thus, carefully chosen soft-

ware design strategies are a prerequisite to tackle signal processing support in ubiquitous contexts [Lazzarini et al. 2012] [Lazzarini et al. 2014].

Ubiquitous-music planning studies involve early assessment of target population expectations and identification of opportunities for creativity support. Through a ubimus planning study, Lima and coauthors (2012)[Lima et al. 2012] found sharply differing expectations on technological usage by musicians and musically naive subjects in educational contexts. Based on these results, they proposed a simple rule of thumb: users like what comes closer to reenacting their previous musical experiences. Non-technical approaches, such as those proposed by traditional soundscape activities [Schafer 1977], may not be suited for introducing non-musicians to sonic composition. Naive subjects may respond better to technologically oriented approaches, as those found in ecologically grounded creative practices [Keller et al. 2014]. If the rule of thumb previously stated holds true, musically naive participants welcome easiness of use and naturality while musicians tend to prefer interfaces that foster behaviors based on acoustic-instrumental metaphors and common-practice music notation. Therefore, design of creatively oriented technologies needs to fulfil different demands depending on the intended user base.

Technological support for pervasive musical activities increases the difficulty of the design task on two fronts. Ubimus systems may enhance the users' creative potential by providing access to previously unavailable material and social resources. But a more intensive usage of resources can introduce unintended complexities, narrowing the access to a small user base. Thus, one challenge faced by ubimus designers is to provide intuitive tools for complex creative tasks. Furthermore, custom-made, special purpose hardware interfaces - such as those proposed by tangible user interface design approaches - may fill the requirements of transparency and naturality reducing the cognitive load of complex tasks. But they do not guarantee wide accessibility. In this case, the catch lies in the financial toll. Special purpose systems are difficult to distribute and maintain. As a consequence, the user base is narrowed by the increased costs of the hardware.

Previous research indicated that another important difficulty faced by the designers of ubiquitous music tools is the slowness of the validation cycle [Keller et al. 2011a]. Because complete integrated systems are hard to design and test, tools usually deal with isolated aspects of musical activity. Musicians usage of the tools may not correspond to the intended design and integration of multiple elements may give rise to unforeseen problems. As a partial solution to these hurdles, the Ubiquitous Music Group has suggested the inclusion of music making within the development cycle. This integration of music making and software development is based on a broad approach to usability [Hornbaek 2006]. Fine-grained technical decisions are done after the usability requirements of the system have been well established through actual usage. So rapid deployment is prioritized over testing on a wide user base. Given the lack of standard support for audio and musical data formats, initial development of audio applications for mobile platforms was feasible but complex and unintuitive [Keller et al. 2010] [Radanovitsck et al. 2011]. Recent advances have paved the way to wider distribution of tools within the computer music community [Brinkmann 2012] [Lazzarini et al. 2012] [Lazzarini et al. 2014]. Within an iterative approach to design - involving creative musical activities and usability assessments - we have developed rapid prototyping techniques tailored for ubiquitous music contexts. Since our research targets both interaction and signal processing, flaws that

arise from coordination among these two processes can be identified early within the design cycle. Furthermore, full-blown creative musical activities uncover opportunities for creative exploration of the software AND of the local resources [Keller et al. 2013]. The prototypes reported in the second part of this paper provide examples of the advantages and limitations of an experimentally grounded approach to the development of ubiquitous music ecosystems.

The last stage of the ubimus design cycle involves the assessment of creative processes and products, targeting the expansion of the creative potentials and the sustainable usage of resources. Although creativity assessment is an active area of research within psychology. [Amabile 1996] assessment of creative outcomes is still a taboo topic among music practitioners. From a product-centered perspective [Boulez 1986], creativity assessment would be equivalent to the measurement of musical value. This approach makes two assumptions. First, the objective of musical activity is to obtain a product that can be labelled as an expression of eminent creativity. Second, the value of the musical product lies in its material constituents (the sounds or their symbolic representation, i.e., scores or recordings of performances). In this case, standards are defined by the adopted compositional technique. Given a technique-centered metric, deviations are seen as spurious, less valuable manifestations. Another problem of the product-centered approach is the overrated reliance on domain-specific expert judgement. When asked to evaluate musical products - as it is done using Amabile's (1996) Consensual Assessment Technique - experts apply socially accepted views on creativity. These views are the result of several years of musical training and experience with eminent forms of creativity. Given the different requirements of professional and non-professional participants [Lima et al. 2012], this bias may render their assessment less useful to everyday-creativity manifestations. To avoid these pitfalls, ubiquitous music projects rely on a mix of assessment techniques [Keller et al. 2011b] engaging a small numbers of expert and untrained subjects in different musical activities in a variety of environmental conditions. This is usually described as 'triangulation' within the behavioral research literature. This approach does not make assumptions regarding the compositional techniques, giving the same weight to musicians' and lay-people's feedback. Data is extracted from the emerging relationships among the user profiles, the activities, the environmental conditions and the support infrastructure.

4. Prototyping platforms for ubiquitous music

During the initial phase of ubiquitous music research (2007-2010), the need of a short development cycle for ubimus infrastructure was faced with multiple obstacles. On one hand, web deployment featured little or no support for audio prototyping. Java and Adobe Flash were the two languages that provided more extensive resources for audio applications [Keller et al. 2011a] [Miletto et al. 2011]. While Java was supported on several mobile platforms, such as JavaME and Android, Adobe Flash was not always available on mobile devices. Hybrid approaches to ubimus system development were introduced involving the use of Javascript and Java-based synthesis engines [Keller et al. 2011b]. An example of this proposal is the ubimus prototyping environment mow3s-ecolab. We describe a case example of the use of mow3s-ecolab in ubiquitous musical activities: the Harpix 1.0 study.

More recently, the development of HTML led to the introduction of audio-oriented web tools. HTML5 features Web Audio and Web Midi JavaScript-based technologies

intended as standards for web deployment. Through the implementation of a ubiquitous music application, we explore some of the potentials and constraints of the Web Audio platform. We describe the development of the LCM Sequencer HTML5, a prototype for the creation of loop-based musical patterns. The design of this ubimus prototype illuminates aspects of the interaction demands of the sequencing activity and highlights the need for accurate timing support for synchronous usage.

As extensively discussed in a recent survey by Wyse and Subramanian (2013)[Wyse and Subramanian 2013], the web browser is now a viable platform for the deployment of music computing applications. Three technologies are dominant in audio development for world-wide web applications: Java, Adobe Flash, and HTML5 Web Audio. Applications based on Java can be rendered either as Applets or via Java Web Start. Adobe Flash has grown in support by multiple browser vendors across various operating systems. Flash applications can be deployed as browser plug-ins, as well as through Adobe Air6. The HTML5 Web Audio framework for Javascript is the newest of these three technologies. Unlike the others, it is a proposed standard that is designed to be implemented by the browser vendors.

Web Audio is today possibly the most popular toolkit for audio development on the web. However it has a number of limitations. Firstly, its set of audio operations is somewhat limited. Its functionality can be extended by Javascript code, which still pays a significant performance penalty if compared to natively-compiled C/C++ code. Although Javascript engines are constantly improving in speed and efficiency, running audio code entirely in Javascript is a processor intensive task on modern systems. However, the worst limitation is that the ScriptProcessorNode which is used to extend the API runs on the main thread. This can result in dropouts when another process on the main thread, for instance the user interface, interrupts or blocks the audio processing. This severely limits what is possible to do with Web Audio in practical terms to the built-in processing nodes. Consequently, we need to look for a technology that allows native applications to do audio processing beyond what it is possible with Javascript and Web Audio. An alternative is provided by the Native Clients (NaCl) platform.

The next three sections present examples of the three approaches just discussed: a prototype using a hybrid Java-JavaScript support system - mow3s-ecolab; an HTML5-based prototype using the Web Audio library; and a sine-wave oscillator exemplifying the usage of the Csound PNaCl programming environment. Each example highlights key requirements of the support for creativity-centered ubiquitous music design involving both musical interaction and audio processing capabilities.

5. Case study: Harpix 1.0

5.1. Interaction patterns and metaphors

The first prototype - Harpix 1.0 - exemplifies the use of the spatial tagging [Keller et al. 2011b]. Spatial tagging is defined as an interaction metaphor that makes use of virtual or material visual cues - anchors - to support creative activity (fig. 1). Anchors provide a bridge between material and cognitive resources, enhancing the creative potential. This approach to the design of ubiquitous music systems has found support in multiple experiments with musicians and non-musicians applying a closely related interaction metaphor: time tagging [Keller et al. 2010] [Keller et al. 2013]

[Pinheiro da Silva et al. 2012] [Pinheiro da Silva et al. 2013] [Radanovitsck et al. 2011] [Pimenta et al. 2012].

Alternatively, Harpix 1.0 can be described as an instantiation of the natural interaction pattern [Flores et al. 2012]. The visual elements of the interface - or anchors - can be manipulated directly, establishing a straightforward relationship between user actions and sound events. This section summarizes the results of an experimental study reported in [Keller et al. 2011b].

5.2. Materials and procedures

MOW3S is a set of tools for multiplatform interface design specifically targeted for web usage. Given the adoption of the standard HTML syntax, MOW3S can be combined with any tool implemented in Javascript. User actions are tracked to generate control data formatted as standard MIDI events which are used to drive the synthesis engine Ecolab. Ecolab is a wavetable synthesis engine implemented in Java. It features support for network connections through standard IP sockets. By adopting DLS and General MIDI standards consistent sonic renditions can be achieved without the need for real-time streaming of audio. Thus, Ecolab can be used as a multiplatform back-end for desktop systems with low computational resources.

Using the mow3s-ecolab environment, Keller and coauthors [Keller et al. 2011b] implemented a prototype based on the spatial tagging metaphor: Harpix 1.0. The Harpix architecture comprises three layers. On the first layer, user interaction is done through text input, mouse position tracking and mouse-wheel movement tracking. The second layer features spatial anchors represented by multiple draggable rectangles distributed on the browser pane. This layer provides synthesis parameter mappings linked to the positions of the anchors on the horizontal and vertical axes. The third layer deals with data routing and sound synthesis.

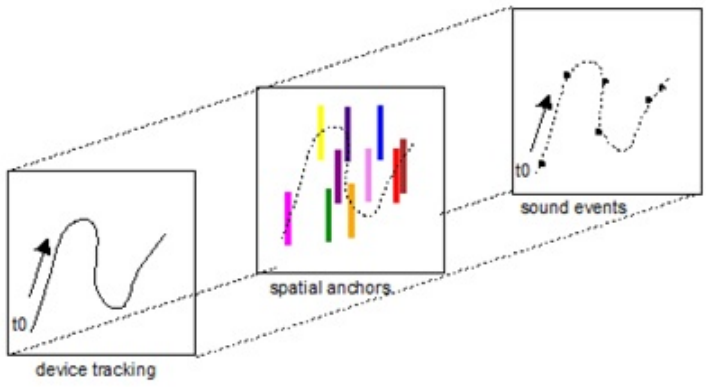

Figure 1. The spatial tagging metaphor in Harpix 1.0.

Three subjects realized 37 interaction essays, comprising multiple conditions (see table 1). The experimenters applied the CSI-NAP v.01 protocol to assess the level of support of the Harpix system for creative musical activities (in a range of 0-10), focusing on six creativity support factors: productivity, expressiveness, explorability, concentration, enjoyment and collaboration. Enjoyment was high during creative (9.5 ± 1.08) and

exploratory activities (8.42 ± 1.78). Expressiveness was also highly rated in creative activities (9.10 ± 0.99). On the other hand, collaboration was poorly judged in all conditions (5.95 ± 2.84).

Activity/Participants	solo	duo	trio	i
creative sessions	4	5	3	12
exploratory sessions	4	5	3	12
imitative sessions	2	10	3	15
i	10	21	6	37

Table 1. Matrix of experimental conditions.

N=3,i=37	productivity	expressiveness	explorability	concentration	enjoyment	collaboration
mean	7.3	7.51	6.08	7.24	7.89	5.95
std. dev.	1.68	2.51	2.54	2.36	2.5	2.84

Table 2. Matrix of experimental conditions.

6. Case study 2: LCM Sequencer HTML5

6.1. Interaction patterns and metaphors

The second prototype provides an example of the application of the event sequencing interaction pattern (figure 2). Multiple loop-based sequences are controlled through a two-tier grid interaction metaphor. On the first tier - selected by clicking the pattern option - each line is assigned a timbre. Columns provide a visual representation of the temporal distribution of the sound events. Color cells indicate events and black cells stand for pauses. Sequence playback is controlled through three GUI elements: the start button, the stop button and the tempo, set as beats-per-minute units. Sound events are rendered through callbacks to the Web Audio synthesis engine. The second tier - available by choosing the song option - provides a preset mechanism. A drag-and-drop mechanism supports direct manipulation of the presets orderings. Up to five presets can be sequenced using the two preset-cell rows.

6.2. Discussion of results

One of the caveats encountered during the preliminary design cycle was the imprecision of the Javascript timer. To circumvent this limitation we resorted to the use of the setTimeOut method, implementing a pooling system with higher resolution for event scheduling. This worked well for stationary platforms but presents occasional problems when running on the Android operating system. The Web Audio timer was accurate at speeds close to 500 BPM. Audio clicks were observed at higher speeds.

Through informal usage testing, we observed that the drag-and-drop preset feature provides an effective shortcut for quick comparisons among multiple complex sequences. This is particularly advantageous when compared to sequencers that are operated through buttons. Rather than manipulating numeric values, the user has direct access to the re-ordering operations. Given that the temporal order of the sequences is correlated to the spatial order of the GUI elements, the anchoring cognitive mechanism furnishes grounding for this interaction metaphor [Keller et al. 2010].

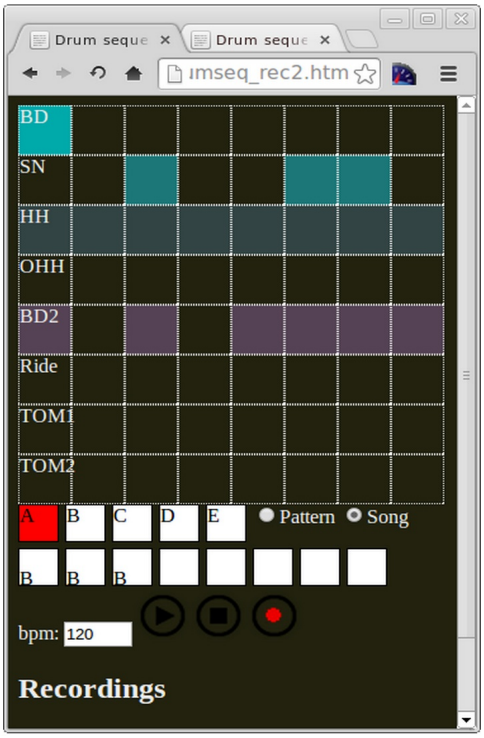

Figure 2. Two-tier grid interaction metaphor for event sequencing: LCM Sequencer HML5.

7. Case study 3: sine-wave oscillator in Csound PNaCl

7.1. Interaction patterns and metaphors

The sine-wave oscillator demonstrates a minimal application of the functionality provided by the Csound PNaCl module. Given the didactic objective of the code, we focused on the use of one controller - represented by an HTML element - and one synthesis parameter - the oscillator's frequency. This is one of the simplest uses of the process control interaction pattern.

7.2. Materials and procedures

The Native Clients (NaCl) platform[1] allows the use of C and C++ code to create components that are accessible to client-side Javascript, and run natively inside the browser. NaCl is described as a sandboxing technology, as it provides a safe environment for code to be executed, in an OS-independent manner [Yee et al. 2009] [Sehr et al. 2010].

The Portable NaCl [Donovan et al. 2010] toolchain, used to implement Csound in this case study, is completely independent of any existing architecture, and thus it is available for a variety of systems. However, PNaCl is currently only currently supported by the Chrome and Chromium browsers (under most operating systems, the iOS and Android versions do not yet support it). Since version 31, Chrome enables PNaCl by default, allowing applications created with that technology to work completely out-of-the-box. PNaCl modules can be served from anywhere in the open web.

[1] `https://developers.google.com/native-client`

The port of the Csound language to the PNaCl platform is complete, apart from its plugin opcodes, which are not available due to non-existence of dynamic loading here. It allows for realtime audio input and output, and it contains a complete Javascript interface that is used to control it. MIDI can be used in the form of MIDI files or through the Javascript implementation (WebMIDI).

7.2.1. Prototype Example

The following script demonstrates a minimal application using the functionality provided by the Csound PNaCl module[2]. It consists of the implementation of the `moduleDidLoad()` callback, where the Csound engine is started (with `csound.Play()` and a simple Csound code is compiled with `csound.CompileOrc()`. This will produce a sine wave whose frequency can be controlled by changing the value of the HTML element with id `freq`:

```
function moduleDidLoad() {
 csound.Play();
 csound.CompileOrc(
 "schedule 1,0,-1 \n" +
 "instr 1 \n" +
 "kfr chnget \"freq\" \n" +
 "a1 oscili 0.5,kfr \n" +
 "outs a1,a1 \n"  +
 "endin");
 SetFreq();
}
function attachListeners() {
  document.getElementById("freq").
      addEventListener("change",SetFreq);
}
function SetFreq(){
 var val = document.getElementById("freq").value;
 csound.SetChannel("freq", val);
}
```

8. Discussion of results

To test the application of the spatial tagging metaphor, our team implemented a prototype based on Java and Javascript browser technology to support creative musical activities: Harpix 1.0. Harpix was used in an experiment encompassing three types of musical activities by three subjects. The assessment of creativity support indicated a high performance in the creative and exploratory activities, with particular emphasis on two factors: enjoyment and expressiveness. However, the collaboration and explorability factors were not evaluated positively. Imitative activities also yielded low scores.

[2] `http://vlazzarini.github.io/docs/pnacl_csound.html`

A second prototype used the HTML5 Web Audio library to support the application of the event sequencing interaction pattern. A two-tier interaction metaphor was developed for synchronous manipulation of complex-sequence orderings. The preset-cells drag-and-drop mechanism showed good potential during preliminary testing, hinting at a common ground for this interaction metaphor, time tagging and spatial tagging [Keller et al. 2010] [Keller et al. 2011b]. The Javascript timer did not perform well. The Web Audio timer performed better, but usage at high metronome speeds produced clicks.

The third prototype explored the facilities provided by the Csound PNaCl environment. We introduced the use of PNaCl for audio programming through a port the Csound language and audio engine. As one of the simplest uses of the process control interaction pattern, the prototype sine-wave HTML-controlled oscillator provided an opportunity to demonstrate the advantages and limitations of this new open-source sand-boxing technology developed as part of the Chromium project. The fully functional implementation of the Csound PNaCl environment features a mid-latency callback mechanism (ca. 10-11 ms, 512 frames at 44.1 or 48 KHz sampling rate) with uniform performance across various platforms. The Audio API design is very straightforward, but it only supports stereo output at one of the two sampling rates just mentioned.

The technologies employed in the development the three prototypes reported in this paper showed different types of limitations for audio programming and interaction support. On one hand, browser-based prototyping, as introduced by the mow3s-ecolab environment, provides a flexible way to deploy and test interaction metaphors. Standard libraries, such as the HTML5 Web Audio and Web Midi, have good potential for wide adoption but currently present design problems that limit their usage in synchronous activities and audio programming tasks. At this point, they are better suited for asynchronous support. We implemented a new set of technologies for audio programming for web applications. The Csound PNaCl environment features a relatively low-latency performance and incorporates the know-how developed over 30 years of Csound usage, providing a path for the development of ubiquitous music ecosystems that goes beyond the HTML5 Web Audio framework.

References

Amabile, T. (1996). *Creativity in Context*. Boulder, CO: Westview Press.

Boulez, P. (1986). *Orientations: Collected Writings*. London, UK: Faber and Faber.

Brinkmann, P. (2012). *Making Musical Apps: Using the Libpd Sound Engine*. O'Reilly & Associates Incorporated.

Bryan, N. J., Herrera, J., Oh, J., and Wang, G. (2010). Momu: A mobile music toolkit. In *Proceedings of NIME 2010*.

Donovan, A., Muth, R., Chen, B., and Sehr, D. (2010). PNaCl: Portable Native Client Executables. *Google White Paper*.

Essl, G. (2010). Urmus: an environment for mobile instrument design and performance. In *Proceedings of ICMC 2010*.

Essl, G. and Rohs, M. (2006). Mobile stk for symbian os. In *Proceedings of ICMC 2006*.

Flores, L., Miletto, E., Pimenta, M., Miranda, E., and Keller, D. (2010). Musical interaction patterns: Communicating computer music knowledge in a multidisciplinary project. In *Proceedings of the 28th ACM International Conference on Design of Communication*, SIGDOC '10, pages 199–206, New York, NY, USA. ACM.

Flores, L. V., Pimenta, M. S., and Keller, D. (2012). Patterns of musical interaction with computing devices. In *Proceedings of the III Ubiquitous Music Workshop (III UbiMus)*, São Paulo, SP, Brazil. Ubiquitous Music Group (g-ubimus), São Paulo, SP, Brazil: Ubiquitous Music Group.

Hornbaek, K. (2006). Current practice in measuring usability: Challenges to usability studies and research. *Internation Journal of Human-Computer Studies*, 64(2):79–102.

Keller, D., Barreiro, D. L., Queiroz, M., and Pimenta, M. S. (2010). Anchoring in ubiquitous musical activities. In *Proceedings of the International Computer Music Conference*, pages 319–326, Ann Arbor, MI: MPublishing, University of Michigan Library. Ann Arbor, MI: MPublishing, University of Michigan Library.

Keller, D., Ferreira da Silva, E., Pinheiro da Silva, F., Lima, M. H., Pimenta, M. S., and Lazzarini, V. (2013). Everyday musical creativity: An exploratory study with vocal percussion (criatividade musicalccotidiana: Um estudo exploratório com sons vocais percussivos). In *Proceedings of the National Association of Music Research and Post-Graduation Congress - ANPPOM (Anais do Congresso da Associação Nacional de Pesquisa e Pós-Graduação em Música - ANPPOM)*. Natal, RN: ANPPOM.

Keller, D., Flores, L. V., Pimenta, M. S., Capasso, A., and Tinajero, P. (2011a). Convergent trends toward ubiquitous music. *Journal of New Music Research*, 40(3):265–276.

Keller, D., Lima, M. H., Pimenta, M. S., and Queiroz, M. (2011b). Assessing musical creativity: material, procedural and contextual dimensions. In *Proceedings of the National Association of Music Research and Post-Graduation Congress - ANPPOM (Anais do Congresso da Associação Nacional de Pesquisa e Pós-Graduação em Música - ANPPOM)*, pages 708–714, Uberlândia, MG: ANPPOM. National Association of Music Research and Post-Graduation (ANPPOM), Uberlândia, MG: ANPPOM.

Keller, D., Otero, N., Pimenta, M. S., Lima, M. H., Johann, M., Costalonga, L., and Lazzarini, V. (2014). Relational properties in interaction aesthetics: The ubiquitous music turn. In *Proceedings of the Electronic Visualisation and the Arts Conference (EVA-London 2014)*. London: Computer Arts Society Specialist Group.

Lazzarini, V., Costello, E., Yi, S., and Fitch, J. (2014). Csound on the web. In *Proceedings of the Linux Audio Conference (LAC2014)*.

Lazzarini, V., Yi, S., Timoney, J., Keller, D., and Pimenta, M. S. (2012). The mobile csound platform. In *Proceedings of the International Computer Music Conference*, pages 163–167, Ljubljana. ICMA, Ann Arbor, MI: MPublishing, University of Michigan Library.

Lima, M. H., Keller, D., Pimenta, M. S., Lazzarini, V., and Miletto, E. M. (2012). Creativity-centred design for ubiquitous musical activities: Two case studies. *Journal of Music, Technology and Education*, 5(2):195–222.

Miletto, E. M., Pimenta, M. S., Bouchet, F., Sansonnet, J.-P., and Keller, D. (2011). Principles for music creation by novices in networked music environments. *Journal of New Music Research*, 40(3):205–216.

Pimenta, M. S., Miletto, E. M., Keller, D., and Flores, L. V. (2012). *Technological support for online communities focusing on music creation: Adopting collaboration, flexibility and multiculturality from Brazilian creativity styles*, volume Cases on Web 2.0 in Developing Countries: Studies on Implementation, Application and Use, chapter 11. Vancouver, BC: IGI Global Press.

Pinheiro da Silva, F., Keller, D., Ferreira da Silva, E., Pimenta, M. S., and Lazzarini, V. (2013). Everyday musical creativity: exploratory study of ubiquitous musical activities (criatividade musical cotidiana: estudo exploratório de atividades musicais ubíquas). *Música Hodie*, 13:64–79.

Pinheiro da Silva, F., Pimenta, M. S., Lazzarini, V., and Keller, D. (2012). Time tagging in its niche: Engagement, explorability and creative attention (a marcação temporal no seu nicho: Engajamento, explorabilidade e atenção criativa). In *Proceedings of the III Ubiquitous Music Workshop (III UbiMus)*, São Paulo, SP, Brazil. Ubiquitous Music Group (g-ubimus), São Paulo, SP: Ubiquitous Music Group.

Radanovitsck, E. A. A., Keller, D., Flores, L. V., Pimenta, M. S., and Queiroz, M. (2011). mixdroid: Time tagging for creative activities (mixdroid: Marcação temporal para atividades criativas). In *Proceedings of the XIII Brazilian Symposium on Computer Music (SBCM)*, Vitória, ES: SBC. Vitória, ES: SBC.

Roberts, C. (2011). Control: Software for End-User Interface Programming and Interactive Performance. In *Proceedings of the ICMC 2011*, Huddersfield, UK.

Roberts, C., Wakefield, G., and Wright, M. (2013). The Web Browser As Synthesizer And Interface. In *Proceedings of the International Conference on New Interfaces for Musical Expression*.

Schafer, R. M. (1977). *The Tuning of the World*. New York, NY: Knopf.

Sehr, D., Muth, R., Biffe, C., Khimenko, V., Pasko, E., Schimpf, K., Yee, B., and Chen, B. (2010). Adapting Software Fault Isolation to Contemporary CPU Architectures. In *19th USENIX Security Symposium*.

Wyse, L. and Subramanian, S. (2013). The Viability of the Web Browser as a Computer Music Platform. *Computer Music Journal*, 37(4):10–23.

Yee, B., Sehr, D., Dardyk, G., Chen, J. B., Muth, R., Ormandy, T., Okasaka, S., Narula, N., and Fullagar, N. (2009). Native Client: A Sandbox for Portable, Untrusted x86 Native Code. In *2009 IEEE Symposium on Security and Privacy*.