Technology and Community in Toolkits for Musical Interface Design

Andrew P. McPherson
Centre for Digital Music
Queen Mary University of London, UK
a.mcpherson@qmul.ac.uk

Fabio Morreale
Centre for Digital Music
Queen Mary University of London, UK
f.morreale@qmul.ac.uk

ABSTRACT

This position paper discusses toolkits for creating digital musical instruments. Musical interaction imposes stringent technical requirements on interactive systems, including high spatial and temporal precision and low latency. Social and community factors also play an important role in musical interface toolkits, including design sharing and the ability of performers and composers to count on the longevity of an instrument. This paper presents three examples of musical interface toolkits, including our own Bela, an open-source embedded platform for ultra-low-latency audio and sensor processing. The paper also discusses how the requirements of specialist musical interface toolkits relate to more general HCI toolkits.

ACM Classification Keywords

H.5.5. Sound and Music Computing: Systems; H.5.1. Multimedia Information Systems: Evaluation/Methodology

Author Keywords

Toolkit, digital musical instrument, embodied interaction, maker community, latency, longevity, pluggable communities.

INTRODUCTION

Musical interaction presents a number of interesting opportunities and challenges for HCI. Many digital musical instruments (DMIs), like their acoustic counterparts, are useful case studies in embodied interaction: extended practice leads to the instrument becoming a transparent extension of the musician's body, where the operations of manipulating the instrument become automatic, allowing the musician to focus on higher-level musical actions [20]. Musical interaction also places stringent technical demands on digital systems, including spatial and temporal precision, high sensor and audio bandwidth, predictability and low latency [7].

Toolkits for creating DMIs have become increasingly common [18, 21, 19, 2, 23, 13, 5], with different projects aimed at a variety of musical contexts and technical skill levels. These toolkits share a desire to enable musicians who are not engineers to create their own high-quality instruments by solving common technical challenges and optimising for the qualities musicians find important.

DMI toolkits provide a common platform for instrument creators to share designs, which serves both research and artistic goals. It takes time for performers to acquire expertise on a new instrument, and encouraging composers to write music for a new instrument requires assurance that the instrument will remain in existence for the piece to continue to be played [15].

In the DMI community, published papers typically contain insufficient detail to fully replicate an instrument design, especially in regard to aesthetic choices and fine details of craftsmanship which are important to the performer experience but might not follow established scientific processes. Some online DMI repositories have been created[1] inspired by more general sharing platforms such as Instructables,[2] however this remains an outstanding challenge for the DMI community. DMI toolkits, by providing a common platform for designers, reduce the barriers to exchanging fully functioning designs.

This position paper explores the current state of musical interface toolkits and their relation to more general HCI toolkits.

[1]For example, Muzhack by Arve Knudsen: https://muzhack.com
[2]http://www.instructables.com

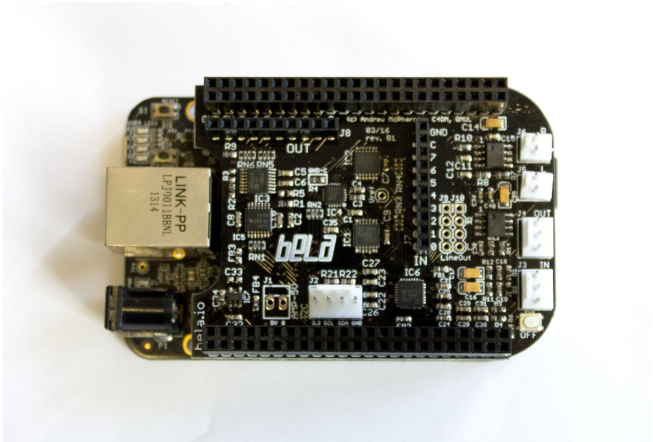

Figure 1. Bela, which consists of a custom hardware board ("cape") on a BeagleBone Black running specialised software.

CHI 2017, May 6-11, 2017, Denver, CO, USA.

ACM ISBN 978-1-4503-4655-9/17/05 ...$15.00.
http://dx.doi.org/10.1145/3025453.3026056

We discuss both technical and social challenges for toolkit design, including the central role of the user community in sustaining a viable toolkit. We then present three established toolkits, including our own Bela[3] (Figure 1), an open-source embedded hardware platform for ultra-low-latency audio and sensor processing [13]. We conclude with suggestions for toolkit designers and topics for discussion at the workshop.

THREE CHALLENGES FOR HCI TOOLKITS

The technical and social challenges we discuss in this section are particularly important for musical interface toolkits, but also apply to HCI toolkits more generally.

Latency

Action-to-sound latency (delay) is a critically important factor in DMI design. Wessel and Wright recommended that DMIs exhibit no more than 10ms latency with no more than 1ms jitter (variation in latency) [24]. An experiment with a digital percussion instrument confirmed these recommendations, showing that performers rated an instrument with consistent 10ms latency the same as one with under 1ms latency, but that 20ms and 10ms $\pm$ 3ms of jitter both rated significantly lower, even when performers did not identify an audible delay [7].

The effects of excess latency include making the instrument feel less responsive, reducing its perceived quality, and potentially disrupting the sensorimotor processes needed for accurate performance. Latency can also be used for deliberate effect: for example, in a multimodal smartphone interface, adding latency to the tactile feedback channel made virtual buttons feel heavier [8].

Surprisingly, 15 years after Wessel's recommendation of less than 10 ± 1ms latency, many commonly used tools for creating DMIs do not meet this standard [14] with jitter posing a particular problem. Achieving low and consistent latency also remains an issue for embodied interaction in other contexts.

Longevity

Many experimental DMIs are designed to be used for only a few performances, but some continue to be used for many years. Once a DMI is created, there is typically little incentive to upgrade its hardware and software, as any change to form or behaviour might disrupt its familiarity to the performer. When DMIs are built using laptops or mobile devices, however, the aim of long-term stability comes into conflict with the need for regular system updates.

Hardware toolkits based on embedded processors (e.g. [2, 13]) provide a potential solution by allowing the instrument to operate standalone without a computer. Ideally, the instrument can be maintained indefinitely on this dedicated hardware. In practice, keeping a DMI toolkit operational over many years remains a challenge. Toolkit design considerations include: high reliability, minimum external hardware or software dependencies, rapid setup time (especially when revisiting an instrument after a long period of disuse) and availability of spare parts. The last consideration points to the value of open-source hardware designs [17], or at least the use of commodity hardware where possible.

[3] http://bela.io

Another question for toolkit designers is whether they seek to support prototyping, extended use, or both. Few mass-market commercial products would be built with HCI toolkits, but at least in the DMI community, it is not uncommon for a toolkit to be used for both prototyping and subsequent production on a scale of dozens or even hundreds of instruments.

Community

The utility of a toolkit cannot be assessed solely by its technical specifications, nor even by the quality of its documentation. An active and cooperative user community also plays a major role in making a toolkit useful to new designers [4, 10]. The success of open-source platforms like Arduino and Processing owes as much to their vibrant online communities as to their engineering features. These communities contribute by publishing example code, providing technical support, creating hardware and software accessories, and helping the original designers maintain the core platforms.

In [17] we explore the process of creating a community around an open-source platform based on our experiences with Bela [13] (described below). We observed that the Bela community grew not only around the intrinsic features of Bela itself, but also through connecting to other established open-source tools. We describe this process as *pluggable communities*: growing a new community in discrete leaps by leveraging established communities around other tools.

THREE MUSICAL INSTRUMENT TOOLKITS

Many musical interface toolkits have been created. The three mentioned here are all open source, publicly available toolkits for creating standalone musical instruments, and all three are still in regular use.

Satellite CCRMA

Satellite CCRMA [2] is a platform for building musical instruments which eliminates the need for a computer. It consists of an ARM Linux distribution with several popular audio programming environments preinstalled, and it is accompanied with a set of example materials for creating instruments [1]. Originally created for the BeagleBoard[4] single-board computer, it has since been released for the popular Raspberry Pi. Instruments built with Satellite CCRMA frequently make use of an Arduino[5] microcontroller board for gathering sensor data, with the Raspberry Pi responsible for audio processing.

Satellite CCRMA is in regular use by DMI designers. Its website[6] links to performances of instruments built with it, and a wiki and online forum provide a means for the community to share knowledge. Its use of the widely available Raspberry Pi, with no dependence on custom hardware, means that the platform itself should be maintainable for years to come, and that software should be easily shared amongst different users. Leaving to the designer decisions on sensors and other external hardware provides significant flexibility, but with the tradeoff of placing responsibility on the designer to maintain and document their own hardware contributions.

[4] http://beagleboard.org
[5] http://arduino.cc
[6] https://ccrma.stanford.edu/~eberdahl/Satellite/

Hoxton OWL

The Hoxton OWL [23] is an open-source programmable audio effects pedal. More recently, it has also been released as a synth module in the popular Eurorack form factor. Like Satellite CCRMA, the OWL is designed for creating musical instruments and audio processing systems, and features a large example library and an active online community.[7]

In contrast to Satellite CCRMA, the OWL is a complete hardware unit based on a custom (though open source) design. Thus, where DMI creators using Satellite CCRMA would likely add their own sensors and other hardware, OWL creators will typically work with the existing controls and focus on software development. Though this reduces the variety of interactive systems that can be created, it makes design sharing especially straightforward. Since the OWL pedal is a self-contained device in a robust stage box, it is likely that any designs running on it can be maintained for many years, though the ability to edit code on the device will remain dependent on a working computer-based compiler toolchain.

Bela

Our lab has created Bela [13] (Figure 1), an embedded platform for ultra-low-latency audio and sensor processing. Bela is based on the BeagleBone Black[8] single-board computer with a custom expansion board ("cape") providing stereo audio I/O with onboard speaker amplifiers, 8 channels each of 16-bit analog I/O and 16 digital I/Os. It uses the Xenomai real-time Linux kernel extensions to process audio and sensor data at higher priority than anything else on the board, including the Linux kernel itself.

The signature feature of Bela is its extremely low latency, under 1ms round-trip for audio or down to 100μs for analog and digital data, with less than 25μs of jitter, outperforming other computer audio environments [14]. It also features an onboard, browser-based IDE with support for C/C++, PureData and SuperCollider programming languages, and an in-browser oscilloscope. Like Satellite CCRMA, Bela is designed for creating self-contained musical instruments, where the designer attaches their own sensor hardware and Bela handles all the computation that would normally be performed by a laptop and a microcontroller board like Arduino.

The platform that later became Bela was originally created for the D-Box, a musical instrument designed to be modified and hacked by the performer [25]. In a CHI 2017 paper [17], we describe the process of developing it from a single-function device to a maker community platform, gradually broadening its scope and improving usability. In April 2016, Bela successfully launched on Kickstarter with the support of over 500 backers. Hardware and open-source design plans are available for sale and download,[9], and we maintain a library of example projects and an online forum[10] for community support and idea exchange.

[7] https://hoxtonowl.com

[8] https://beagleboard.org/black

[9] https://shop.bela.io and http://bela.io/code, respectively

[10] http://forum.bela.io

OUR RELATED WORK

The Augmented Instruments Laboratory,[11] led by the first author, is a research team within the Centre for Digital Music at Queen Mary University of London. An augmented instrument is a traditional musical instrument whose capabilities have been technologically extended, maintaining the familiarity and cultural connotations of the original instrument while extending its capabilities.

In addition to Bela [13], described in the preceding section, our previous projects include several augmented instruments including the magnetic resonator piano [15], an electromagnetically-actuated acoustic grand piano and TouchKeys [12], a sensor kit adding multi-touch sensing to the surface of the piano keyboard. Our research also encompasses studies of performer-instrument interaction in solo [7] and group [16] settings and studies of audience perception of performance [3].

CONCLUSION: POSITION ON TOOLKITS

As creators of an open-source DMI toolkit, we are especially interested in the potential for toolkits to broaden access to interactive system design. HCI toolkits, including those for creating musical interfaces, contribute to and benefit from larger trends in maker culture [9]. Here we set out two specific arguments for possible discussion at the workshop.

Toolkits need a two-way dialogue with their communities

User-centred and participatory design methodologies have long histories in HCI, so to say that a toolkit should respond to the needs of its community borders on cliché. In fact, we would argue that there is a risk in being *too* reactive to perceived user requirements. It has long been observed that people use technology in unexpected ways, and this process of appropriation has influenced HCI design methods [6]. Similarly, the history of music is replete with examples of people playing instruments in unexpected ways [25]. Performers, upon encountering a new instrument, explore its creative opportunities and constraints [11], but they should not be expected to imagine hypothetical capabilities of instruments that do not yet exist [15].

We argue that HCI toolkits, while being sensitive to community needs, should also express the creative intentions of the toolkit designer. This way, the designer not only contributes new ideas back to the community, they also provide unique "signature features" that may improve the uptake of their tools [17].

No toolkit is aesthetically neutral

Every musical instrument encourages certain possibilities while discouraging others. Some constraints are obvious: the piano can only play 88 discrete notes, and does not allow the performer to shape them after they are struck. Others are less obvious: patterns of notes in piano music are typically those which fit the shape of the hand, which are different than the patterns likely to be convenient on a wind or string instrument. Similarly, even if two DMIs can control identical dimensions of the sound, their differing physical designs might encourage different choices of actions.

[11] http://www.eecs.qmul.ac.uk/~andrewm

Tuuri et al. [22] distinguish between *push effects* which force or guide the user to particular choices, versus *pull effects* which relates to the ease of conceiving how an action relates to an output. Music programming languages, supposedly able to create any sound, may nonetheless exhibit strong pull effects by making certain structures and actions easier than others. For that reason we might speculate that every computer music language has its own signature sound.

More broadly, we would argue that toolkit designers can and should embrace the aesthetic influence of their toolkits. A good toolkit might allow the creation of many different types of systems with widely varying aesthetics, but certain possibilities will always be more obvious than others. Rather than striving for an elusive neutrality, toolkit creators might do best to acknowledge their own personal outlook and the influence it is likely to have on designers and end users.

ACKNOWLEDGEMENTS

This work was funded by EPSRC under grant EP/N005112/1 (Design for Virtuosity: Modelling and Supporting Expertise in Digital Musical Interaction).

REFERENCES

1. Edgar Berdahl. 2014. How to Make Embedded Acoustic Instruments.. In *Proc. NIME*.
2. Edgar Berdahl and Wendy Ju. 2011. Satellite CCRMA: A Musical Interaction and Sound Synthesis Platform.. In *Proc. NIME*.
3. S Astrid Bin, Nick Bryan-Kinns, and Andrew McPherson. 2016. Skip the Pre-Concert Demo: How Technical Familiarity and Musical Style Affect Audience Response. In *Proc. NIME*.
4. Leah Buechley and Benjamin Mako Hill. 2010. LilyPad in the wild: how hardware's long tail is supporting new engineering and design communities. In *Proceedings of the 8th ACM Conference on Designing Interactive Systems*. ACM, 199–207.
5. Filipe Calegario, Marcelo M Wanderley, Stéphane Huot, Giordano Cabral, and Geber Ramalho. 2017. A Method and Toolkit for Digital Musical Instruments: Generating Ideas and Prototypes. *IEEE MultiMedia* 24, 1 (2017).
6. A. Dix. 2007. Designing for appropriation. In *Proc. British HCI Group Conf. on People and Computers*.
7. Robert H Jack, Tony Stockman, and Andrew McPherson. 2016. Effect of latency on performer interaction and subjective quality assessment of a digital musical instrument. In *Proceedings of the Audio Mostly 2016*.
8. Topi Kaaresoja and Stephen Brewster. 2010. Feedback is... late: measuring multimodal delays in mobile device touchscreen interaction. In *International Conference on Multimodal Interfaces*.
9. Stacey Kuznetsov and Eric Paulos. 2010. Rise of the expert amateur: DIY projects, communities, and cultures. In *Proc. NordiCHI*.
10. Silvia Lindtner, Garnet D Hertz, and Paul Dourish. 2014. Emerging sites of HCI innovation: hackerspaces, hardware startups & incubators. In *Proc. CHI*.
11. T. Magnusson. 2010. Designing Constraints: Composing and Performing with Digital Musical Systems. *Computer Music J.* 34 (2010), 62–73. Issue 4.
12. A. McPherson, A. Gierakowski, and A. Stark. 2013. The space between the notes: adding expressive pitch control to the piano keyboard. In *Proc. CHI*.
13. Andrew McPherson and Victor Zappi. 2015. An environment for submillisecond-latency audio and sensor processing on BeagleBone Black. In *Proc. AES 138th Conv.*
14. Andrew P McPherson, Robert H Jack, Giulio Moro, and others. 2016. Action-Sound Latency: Are Our Tools Fast Enough?. In *Proc. NIME*.
15. Andrew P McPherson and Youngmoo E Kim. 2012. The problem of the second performer: Building a community around an augmented piano. *Computer Music Journal* 36, 4 (2012), 10–27.
16. Fabio Morreale, Antonella De Angeli, Raul Masu, Paolo Rota, and Nicola Conci. 2014. Collaborative creativity: The music room. *Personal and Ubiquitous Computing* 18, 5 (2014).
17. Fabio Morreale, Giulio Moro, Alan Chamberlain, Steve Benford, and Andrew P. McPherson. 2017. Building a Maker Community Around an Open Hardware Platform. In *Proc. CHI*.
18. Axel Mulder. 1995. The I-Cube system: moving towards sensor technology for artists. In *Proc. of the Sixth Symposium on Electronic Arts (ISEA 95)*.
19. D. Newton and M. T. Marshall. 2011. Examining How Musicians Create Augmented Musical Instruments. In *Proc. NIME*.
20. Luc Nijs, Micheline Lesaffre, and Marc Leman. 2009. The musical instrument as a natural extension of the musician. In *Proc. Interdisciplinary Musicology*.
21. Dan Overholt. 2006. Musical interaction design with the Create USB interface. In *Proc. ICMC*.
22. Kai Tuuri, Jaana Parviainen, and Antti Pirhonen. 2017. Who Controls Who? Embodied Control Within Human–Technology Choreographies. *Interacting with Computers* (2017).
23. Thomas Webster, Guillaume LeNost, and Martin Klang. 2014. The OWL programmable stage effects pedal: Revising the concept of the on-stage computer for live music performance.. In *Proc. NIME*.
24. D. Wessel and M. Wright. 2002. Problems and Prospects for Intimate Musical Control of Computers. *Computer Music Journal* 26, 3 (2002), 11–22.
25. Victor Zappi and Andrew McPherson. 2014. Design and Use of a Hackable Digital Instrument. In *Proc. Live Interfaces*.