Making Sound Synthesis Accessible to Children

Christoph Trappe
Center for Computing and Communication Technologies (TZI)
Bibliothekstr. 1
28359 Bremen, Germany
ctrappe@tzi.de

ABSTRACT

In this paper we present our project to make sound synthesis and music controller construction accessible to children in a technology design workshop. We present the work we have carried out to develop a graphical user interface, and give account of the workshop we conducted in collaboration with a local primary school. Our results indicate that the production of audio events by means of digital synthesis and algorithmic composition provides a rich and interesting field to be discovered for pedagogical workshops taking a Constructionist approach.

Keywords

Child Computer Interaction, Constructionism, Sound and Music Computing, Human-Computer Interface Design, Music Composition and Generation, Interactive Audio Systems, Technology Design Activities.

1. INTRODUCTION

Music constitutes a significant part of children's lives. While the construction of novel musical interfaces enjoys great popularity amongst professionals such as engineers, scientists, and artists, we argue that the educational potential of the interface creation has been neglected in the past. By taking advantage of this potential in an educational setting, music can nurture interest in active participation in the design and development of technological artifacts [16]. Sound synthesis can provide a unique access to technology and hence foster interest in Information and Communications Technology (ICT) related subjects. This requires appropriate interfaces and workshop concepts. A Constructionist [9] sound environment could serve as a rich field to generate a personal and meaningful understanding of the creative possibilities of computer generated music. Although some prior work suggest that music has a great potential to foster children's interest in technology and some stress on its supportive qualities in learning processes, the community lacks appropriate sound design environments for children. Core issue here is the software for children to program and build interesting sounds. Children should find themselves as creators of technology and active participants in the design process, rather than mere consumers or spectators. With our project we would like to address this problem. With a new sound programming interface we aim to augment existing smart textile workshop concepts by the realm of interactive music creation. In this paper we present our work on the derivation and development of a graphical user interface (GUI) to enable children to program simple sound synthesis patches to be controlled through sensors such as accelerometers or light sensors. Furthermore we sketch its use in a technology design workshop.

NIME'12, May 21 – 23, 2012, University of Michigan, Ann Arbor.

2. RELATED WORK

Constructionist [9] workshops with smart textiles [11] and construction kits such as the EduWear kit [12] or the LilyPad platform [3] have shown that children have considerable interest in embedding sound capabilities in their artifacts. Generally the standard workshop set-up provides a good basis for the creation of personal music controllers, since a huge variety of materials, tools and components are available [12]. As stated previously, the possible ways to create sensor interfaces with self-made sensors for music interaction provide a vast area to be explored, for instance [8] (the mentioning "homemade pressure and position sensors", sensors from video tape or antistatic foam), [6] ("conductive ink, adhesive, rubber, tape, elastics and porous materials"), [4] (pencil lead resistors and homemade tilt switches, to name but a few ways to interface to music). Software environments specially designed to develop programs that synthesize sounds are very abstract and require a rather large knowledge base to formulate musical expressions. This is true for text based programming environments such as SuperCollider or ChucK [18] as well as for graphical programming environments such as *Pure Data* [10] or MaxMSP. A Sound Modelling and Control Kit (SMaCK) geared for children aged 9-13 was presented in [17]. Children can gain access to additive sound synthesis and control their sound with a personal hardware controller. However, SMaCK lacks programmability and provides merely linear assignment of parameters to sound properties.

3. PRE-STUDY

In the following section we present our project work. Starting with the motivation, we will sketch our development process from the first pre-test with ideas of participatory design [19] to the working prototype used in the final school workshop. Within a Halloween Theme Based Workshop children were assisted in using the Arduino board to trigger simple pre-recorded wave sounds. While this enabled the participants to create fun artifacts it neither provided a deeper understanding of digital sound production nor enabled further interaction with the sound itself. However, this initial set up convinced us that the children were interested in creating more flexible sounds. The next step was to experiment with existing professional environments in order

to enable the creation of more powerful artifacts in terms of sound capabilities. We conducted a test with a small group of children (four participants) using *Pure Data* in order to understand the potential of a unit generator based graphical programming environment and the problems which may arise during the interaction with it. This approach was inspired using methods of participatory design which involved the users at an early stage of development, rather than just evaluating the finished product at the very end, where feedback has little or no impact. The approach required constant iteration of the software evolving along user feedback. A graphical programming environment for sound synthesis was chosen for an initial study with children to gather material on how sound programming could be made accessible. Several projects revealed that graphical programming languages have great potential to be accepted and understood by the youngsters. Popular examples for successful environments are the LEGO-Mindstorms programming environment *RCX-Code*, *Scratch*[1], or the Java based *Amici*[2] (to program the Arduino board). Over a period of six weeks four children (aged 11-14) visited the university once a week to work with *Pure Data* in combination with other craft materials. Over this period different approaches to the software were taken. The general notion was to move from an empty patch to a pre-configured interface with custom high level abstractions (e.g. frequency modulation). In the beginning a small number of intrinsic elements of *Pure Data*[3] was explained to the children, such as osc~, dac~, metro and random. The sessions were captured by means of video and audio recordings. The two tutors were asked to write a brief protocol after each session. They were asked to observe the children in the process of constructing sound pieces. One focus was to pay attention to "indicators of enjoyment and engagement" as defined in [15]: "smiles, laughter, or positive body language, and also signs of lack of enjoyment and frustration, including, for example sighs, and looking around the room". The documentation should include the process of artifact design, referring to the actual behavior of participants (social interaction, expressions) and their activities (planning, constructing, playing, etc.) as well as the construction process in hardware and software. During the session the children were encouraged to give their opinion, to discuss problems, exchange ideas, communicate critical parts of the software, present their work, reflect on it and perform together. Based on the observations and the feedback of the children the interface was altered and extended by the tutors consecutively from session to session, resulting in a color coded pre-configured patch with custom high level abstractions. Most handling issues observed were related to the GUI. We can report a significant amount of positive feedback from the subjective user experience of the participants that used the software. The fundamental principal of graphical programming, connecting units to form a patch, was conceived as stimulating. It comes as no surprise that the raw PD GUI provided some difficulties regarding usability. We have summarized the main difficulties of the children using the software as follows:

- Elements are not easily accessible. One has to know the name of an object to use it.
- Object names are not expressive. Without prior knowledge identifiers such as vcf~ do not mean much.
- Language issues (only English version).
- The interface is based on the drag-and-drop metaphor. The handles of objects are too small, therefore they are hard to grab.
- There are only few visual cues on the functionality of a patch (e.g. the order of connections is invisible, hot and cold inlets look the same).
- It is unclear why connections cannot be drawn from *inlet* to *outlet* (from *drain* to *source*).

We figured that most of the shortcomings that became apparent during the pre-test could be eliminated by providing an alternative GUI. Based on the findings and the experiences made during this phase we developed a GUI, which is presented in the next section.

4. GUI PROTOTYPE

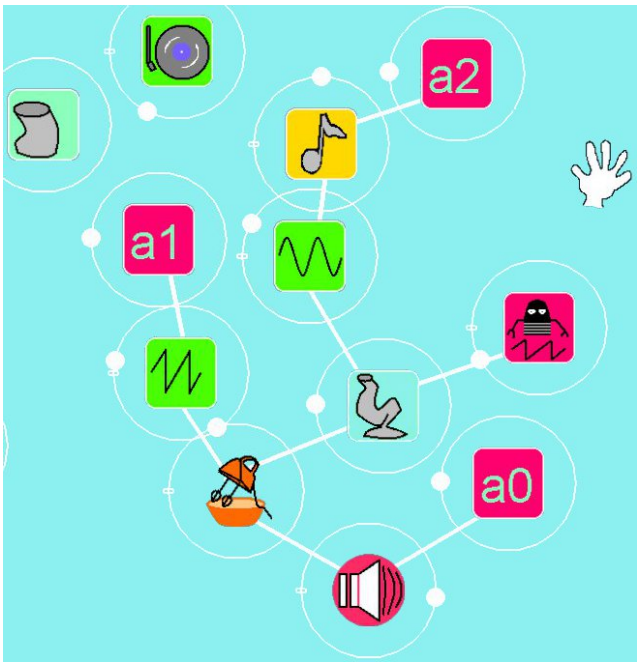

Figure 1: GUI showing a childs patch with three analog inputs coming from the Arduino (a0, a1, a2) controlling master volume, saw tooth frequency and sine wave pitch.

As a consequence of the pre-test we decided to replace the textual identifiers of unit generators and filters with distinct iconic representations. At the start of the program the objects are already present on the screen. Objects can be dragged and dropped - without prior navigation through multi-layered menus. Since some difficulties arose when making explicit connections between objects, we reviewed literature for alternatives. As appropriate for this purpose we identified the dynamic patching [7] approach as used for the reacTable [14]. The alternative GUI (Figure 1) was designed to have a small number of elements such as unit generators, controllers and filters that were refined in the color coded pre-configured patch during the pre-test with the children. Furthermore we provided objects to represent data input from the Arduino board. The GUI was implemented using the open source C++ toolkit *openFrameworks*[4]. *Pure Data* was used as the sound engine,

[1]Developed by the Lifelong Kindergarten Group at the MIT Media Lab, `http://scratch.mit.edu/`
[2]`http://dimeb.informatik.uni-bremen.de/eduwear/`
[3]Pd version 0.41.4-extended
[4]`www.openframeworks.cc`

receiving OSC messages from the GUI. The serial connection between microcontroller and host computer was realized with the Firmata[5] standard firmware and the Pduino object[6]. On the hardware side the connection was realized with Xbee modules[7].

5. SCHOOL WORKSHOP AND RESULTS

The implemented GUI was then to be used in a sound workshop. The workshop "Moving Sounds" was organized in collaboration with a local primary school. 20 girls and boys of the 4th grade were involved in inventing, creating, performing music controllers twice a week over a period of five weeks with a public performance at the end to present their inventions (Figure 2). The sessions were planned

Figure 2: Children rehearsing gestures for the performance with their digital sound artifacts.

to follow the five pedagogical stages as described in [5]: 1. During the first session children had the opportunity to freely think about what kind of sounds they would like to make. They collected ideas and sketched first blueprints loosely related to fantastic musical instruments. At this stage they were not confronted with the constraints of the hardware and software. 2. The children were introduced to sensors, actuators and the microcontroller. The second stage was also used to introduce the children to the GUI for sound production. 3. The children formed project groups to combine ideas from the first sessions with the knowledge gained during the second one. The children reinvented their ideas with their acquired technical knowledge and worked in groups on realizing them. 4. At the fourth stage the children could follow their ideas and complete some hands-on work with constructing and programming. 5. The public presentation was an important part of the pedagogical set-up. The children could show what they had learned and how they gained confidence in the interaction with the technology.

The GUI was used with an enthusiasm that actually became a usability issue; for practical reasons the children worked on a project in groups of four. This proved to be challenging at different stages, for instance while working with the software, as every group member wanted to be in charge of the mouse. The individuals had no difficulty in creating patches and the interaction with the GUI was perceived by the three workshop tutors as enjoyable for the children.

Within the five weeks five sound artifacts were created. They are shown in Figure 3. The first three seem to be quite similar, in that they are all shirt based. However, the theme they followed and the sound that was programmed for them using the GUI, differed to a large extent. The first one, the ZOMBIE SHIRT, was meant to produce a creepy, eerie sound controlled through a self-made pressure sensor. The second one, the BREAKDANCE SHIRT, used accelerometers to speed up and slow down a drum loop to produce "the soundtrack to the dance" on-the-fly. The third shirt was created to be a percussion instrument; two pressure sensors could trigger short harmonic sounds. Number four in Figure 3 shows the ARM FLUTE. A temperature sensor in a straw was used to probe for the musician blowing. The sensor values were mapped through the GUI onto the amplitude of a melody, with speed controlled by means of a pressure sensor. The final artifact number five was the only creation not mounted to the body. The so called STOMP PIANO was placed on the stage and controlled through self-made pressure sensors to vary pitch and volume of the sound.

Figure 3: Childrens inventions: 1. Sound Zombie, 2. Breakdance Shirt, 3. Body Drum, 4. Arm Flut, 5. Stomp Piano.

6. CONCLUSION

The workshop was discussed by the tutors and pupils at the end of each session as well as during a final meeting after the performance. The overall feedback was very positive. However, during the performance it became apparent that the division of a music controller in sound generating unit (the computer) and the controller (the artifacts) was problematic from an experience design point of view. Sometimes it was confusing for the little performers. For future workshop we suggest to mount the sound producer on the artifact itself, using for instance battery powered PC speakers. We assume that the children would have felt more in control over the instrument. In this we would suggest to aim to create AutoNMAs as coined in [1] denoting *autonomous new media artifacts*. We strongly recommend to make the artifacts self-contained - also to avoid the problematic wireless connection. This might be achieved through mobile-phone centered workshops. Another approach to this could be to use the microcontroller itself for sound production by toggling I/O pins at a certain rate (aka *bit-banging*) or by using a simple DAC and multiple output pins[8]. By implementing a bit-banging interface for children the artifacts could become independent from a host computer. The digital processing could be linked closer to the sound generation.

[5] `www.firmata.org`
[6] `at.or.at/hans/pd/objects.html#pduino`
[7] `www.digi.com`
[8] `http://www.uchobby.com/index.php/2007/11/11/arduino-sound-part-1/`

Bringing the sound production closer to the actual events on the hardware level one might also avoid what Hejlsberg refers to as *simplexity* [2]. The exploration of fundamental programming concepts such as control statements (e.g. relational operators or conditionals) could be enabled. For the implementation of sound synthesis software for children we suggest to focus on *musical playfulness* as a measure for a successful design. Play is a significant strategy to acquire knowledge about the world. We take the diversity of artifacts that were created during the workshop as an indicator that the overall concept is not restrictive and indeed enables children to realize their ideas in a free flow. Two key words are especially important when designing for children: simplicity and robustness. The first one was also stated in [13]. It is important that the children feel in control and are independent from a teacher or a tutor. Robustness is an equally straight forward point, however worth mentioning here. The goal should be to provide a stable environment that does not impose interruptions due to necessary software debugging in the workshop. From the observations it is possible to draw a number of conclusions. Apart from some usability issues with the current environment (as identified in the pre-test in Section 3) it turned out that children are motivated to dedicate their attention to a rather abstract environment that offers an interesting, rewarding output with aesthetic qualities. By trend the sound environment nurtures intrinsic motivation and enables self-motivated creation, exploration, and play. The iconic representations as used in the GUI are still abstract; yet they seemed to offer more room for association than just the textual identifiers. This could be extended by enabling the participants to design personal icons for each object. To combine musical instrument construction and sound programming for children in a workshop is very promising in terms of the variety of activities that can be offered in such a context, ranging from the action (during the construction), interaction (with technology and sound), and acting (during the performance). The development of such interfaces is an under-researched field. The huge variety of controllers that came into existence in the last couple of years provides an indication of the huge variety of possible modalities that can be addressed with different materials. Hence music controller construction is a rich field to be discovered for pedagogical purposes. An approach to acquire some fundamental understanding of how the current data flow programming environment *Pure Data* could be modified to be utilized in a workshop setting was presented on the preceding pages.

7. ACKNOWLEDGMENTS

We are grateful for the funding provided by the *Klaus Tschira Stiftung* in the scope of the Graduate School "Advances in Digital Media" at the University of Bremen. Furthermore we would like to thank the communities of *openFrameworks*, *Pure Data*, and *Arduino*. We would also like to thank the workshop participants for their valuable feedback.

8. REFERENCES

[1] E. Berdahl and C. Chafe. Autonomous new media artefacts (AutoNMA). In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 322—323, Oslo, Norway, 2011.

[2] F. Biancuzzi and S. Warden. *Masterminds of Programming: Conversations with Creators of Major Programming Languages*. O'Reilly Media, 1 edition, Apr. 2009.

[3] L. Buechley and M. Eisenberg. The LilyPad arduino: Toward wearable engineering for everyone. *IEEE Pervasive Computing*, 7(2):12–15, 2008.

[4] N. Collins. *Handmade Electronic Music: The Art of Hardware Hacking*. Taylor & Francis, 2nd ed. edition, June 2009.

[5] N. Dittert, K. Dittmann, T. Grüter, A. Kümmel, A. Osterloh, M. Reichel, H. Schelhowe, G. Volkmann, and I. Zorn. Understanding digital media by constructing intelligent artefacts - design of a learning environment for children. In *ED-MEDIA World Conference on Educational Multimedia, Hypermedia & Telecommunications*, pages 2348–2358, Chesapeake (VA), 2008. AACE.

[6] A. Jensenius, R. Koehly, and M. Wanderley. Building Low-Cost music controllers. In *Computer Music Modeling and Retrieval*, pages 123–129. 2006.

[7] M. Kaltenbrunner, G. Geiger, and S. Jordà. Dynamic patches for live musical performance. In *Proceedings of the 4th Conference on New Interfaces for Musical Expression*, Hamamatsu (Japan), 2004.

[8] E. R. Miranda and M. M. Wanderley. *New Digital Musical Instruments: Control And Interaction Beyond the Keyboard*. A-R Editions, Pap/Com edition, July 2006.

[9] S. A. Papert and I. Harel. Situating constructionism. In *Constructionism*. Ablex Publishing Corporation, 1991.

[10] M. S. Puckette. Pure data: another integrated computer music environment. In *Proceedings Second Intercollege Computer Music Concerts*, pages 37–41, Tachikawa, 1996.

[11] M. Reichel, L. Dickmann, and Schelhowe. Smart textiles prototyping: Participating with new ideas. In *CARS2007*, 2007.

[12] M. Reichel, A. Osterloh, E. Katterfeldt, D. Butler, and H. Schelhowe. EduWear: designing smart textiles for playful learning. In *Proceedings of the 8th ICICTE*, volume Readings in Technology in Education, pages 252–263, Corfu, Greece, 2008.

[13] M. Resnick and B. Silverman. Some reflections on designing construction kits for kids. In *Proceedings of the 2005 conference on Interaction design and children*, pages 117–122, Boulder, Colorado, 2005. ACM.

[14] Sergi Jordà, M. Kaltenbrunner, G. Geiger, and R. Bencina. The reacTable. In *Proceedings of the International Computer Music Conference*, Barcelona (Spain), 2005.

[15] G. Sim, S. MacFarlane, and J. Read. *All work and no play: Measuring fun, usability, and learning in software for children [An article from: Computers & Education]*. Elsevier, Apr. 2006.

[16] A. Thaler and I. Zorn. Music as a vehicle to encourage girls' and boys' interest in technology. Mar. 2009.

[17] C. Trappe and H. Schelhowe. SMaCK: a sound modelling and control kit for children. In S. Kain, D. Struve, and H. Wandke, editors, *Workshop-Proceedings of the 2009 Conference on Humans and Computers*, pages 183–186, Berlin, 2009. Logos Berlin.

[18] G. Wang and P. Cook. ChucK: a concurrent, on-the-fly, audio programming language. In *In Proceedings of the ICMC*, 2003.

[19] B. Williamson. The participation of children in the design of new technology - a discussion paper, Sept. 2003.