JMTE 5 (2) pp. 181–193 Intellect Limited 2012

Journal of Music, Technology & Education
Volume 5 Number 2
 doi: 10.1386/jmte.5.2.181_1

HANS ROELS
University College Ghent

Abunch, a tool to teach live electronics in pre-college music education

ABSTRACT

This article[1] proposes an outline for a live electronics course in pre-college music education, examines whether open source music software is suited to teach live electronics and finally presents Abunch, a library in Pure Data created by the author, as a solution for the potential educational disadvantages of open-source music software.

KEYWORDS

live electronics
music education
Pure Data
digital musical instruments
open-source music software
music technology

1 INTRODUCTION

For more than a decade, home computers and laptops have become powerful enough to process sound in real time. This has transformed computers into musical instruments. As this change was taking place, computers became more widely available in households and schools. By 2011, the computer has probably become one of the most widespread musical instruments in much of the developed world.[2]

This unique situation prompts us to rethink and redesign our pre-college[3] music education in which performing electronic music still plays a minor role (Pestova 2009: 124). Books such as *Technology Integration* (Burns 2008) and *Strategies for Teaching Technology* (Reese et al. 2001) offer many strategies and lessons to integrate technology in music education. More than 50 teachers

1. An earlier version of this article was published in the Proceedings of the Linux Audio Conference 2010.

2. According to Eurostat (2011), 60% of the households in the European Union had access to a personal computer in 2006. In the United Kingdom,

this percentage has risen to 77% in 2010 (Office for National Statistics 2011). These figures are higher in households with children. In 2008–2009 91% of households with children had access to a home computer in Australia, compared to 73% in households without children (Australian Bureau of Statistics 2011). In Belgium, this was 87% compared to 65% in 2009 (Statbel 2011). Figures on the number of music instruments per household are not known to the author.

3. The term 'pre-college music education' is used to denote all kinds of music education for children, teenagers and adults who have not taken music courses on a college, university or professional level.

4. Live electronics is used in a broad sense to describe a public performance with at least a human performer and an electronic device producing or processing sound.

5. Abunch is available for download at www.hansroels.be/abunch.htm. It is released under the Creative Commons GNU General Public Licence.

6. Music schools in Belgium – more specifically in the region of Flanders – are state-subsidized schools that teach music, theatre and dance outside the regular day schools.

submitted material for the two books, but almost all included strategies related to performing electronic music, prescribe the use of standard hardware instruments such as General MIDI-compatible synthesizers or digital pianos. The potential of the computer as a tool for performing live music is hardly ever discussed. Therefore, the remark made by Jonathan Savage (2009: 156) still remains relevant: 'Designing, developing and prioritising a performance practice for digital music performance tools and situating this within an educationally rich performance ecology seem like essential tasks for future research and development'.

This article proposes an outline for a live electronics[4] course on a pre-college level, examines whether open-source music software – specifically Pure Data (Pd) – is suitable as a tool for such a course. It presents Abunch[5], a library in Pd created by the author, as a solution to the potential educational disadvantages of open-source music software.

The origin of the outline and the software library discussed in this article lie in an educational project performed by the author from 2007 to 2010 in a music school in Deinze (Belgium).[6] In this project, teenagers attended a weekly two-hour lesson in which they learned how to perform live electronic music. They could follow such a course for one, two or even three years. This long duration gave the author the opportunity to develop long-term goals for the project and experiment with teaching live electronics beyond an introductory stage in which pupils were only introduced to new interfaces and software instruments. As Oore (2005: 64) notes when talking about learning to play new musical interfaces: 'The introductory stages are but a small part of it – the true creative journey begins when the user's own goals and style drive the learning, and when basic elements begin to be internalized and built upon'.

2 AN OUTLINE FOR A LIVE ELECTRONICS COURSE

2.1 Digital musical instruments

It is important to know the fundamental differences between live electronic and acoustical music before the content and philosophy of a live electronics course can be discussed.

First, Simon Emmerson (2007: 198) observes that 'the first synthesis removed the need for the mechanical causality of sound'. An electronic piano sound does not need to be produced by the movement of a hammer against a string. In a digital musical instrument (DMI), the sound production unit and the user interface can be separated and recombined (Miranda and Wanderley 2006). It is possible to play piano sounds by blowing on a wind controller. Although a designer can hide the modular nature of a DMI by hardwiring the connection between a specific user interface and a sound synthesis technique in one device, e.g. a digital piano, the computer and its software have the potential to bring this modularity to the fore (Jordà 2008: 95). In most computer software for live electronic music, several devices can be used as user interface and the data from these interfaces can be assigned to different functions within the software. It is the modular nature that distinguishes a DMI from other traditional instruments, and therefore this unique modularity is a leading principle in the proposal of a live electronics course and of the Abunch software library.

Second, what does performing well mean in a live electronic context, as the goal of a live electronics course is to teach children and other students to

play this instrument well. Virtuosity in the digital musical era has an offstage component that is almost as important as the onstage one and which is not only different but also more diverse and extensive than in acoustical virtuosity. Developing, building, modifying and becoming familiar with a digital instrument or a set-up is highly important for producing a convincing and expressive performance apart from the classical training (Brown 2007: 116, 135). Live electronic music often includes the operation of prosthetic elements such as pedals, keyboards, sensors and other devices; therefore, McNutt (2003: 299) concludes that 'Practising with the equipment is therefore every bit as important as practising with the score'.

The third difference can be found in the information stream between composers and performers. In electronic music, the number of parameters that can be manipulated (before or during a performance) turn a traditional score into a restricted medium to communicate a message between a composer and a performer (Eimert 1957). On top of this, Gariépy and DéCarie (1984: 2) state that 'electro-acoustic music arose at a time when the notion of musical notation itself was the subject of profound questioning'. Therefore, the role of the score in live electronic music is minimal, and Gregorio García Karman (2012) even calls the performance of electro-acoustic music 'a practice in which the score is not a necessary condition at any of its stages of production'. If a performer wants to learn from a composer or from other performers how to perform, he has to attend a performance, talk directly to a colleague, technician, or composer, or consult other media (recordings, texts, source codes, patches, websites, mailing lists, videos), more than just look at a score. Information has become multimodal in live electronics.

Live electronic music is clearly different from acoustical music, and its categories of human activities reflect these changes: the boundaries between composer, improviser, performer, technician and instrument-builder (McNutt 2003: 302) have been blurred, and performing has in fact become an inadequate term. In general, whenever this term is used in live electronics, a larger amount of composition, improvisation and instrument-building is implied than in acoustical music.

2.2 Content

Taking into account the specific character of the musicianship and the instrument in live electronics, a basic content of a beginner's course for live electronic music is presented. In short, the following knowledge domains are part of this content:

1. digital signal processing techniques
2. basic audio hardware
3. mapping techniques
4. history of electronic music
5. auditory training
6. sound organization in real time (improvisation)
7. performance training.

Each of these seven domains is very extensive and could be the subject of a new course. In a course of live electronics on a pre-college level, these subjects need to be treated only very basically, and a selection within each domain has to be made. The separate implementation in music education of six of

these domains is not new and has been researched and applied in classrooms before, in courses or areas of competency such as electronic music instruments, music production (Rudolph et al. 2005: 4) or synthesizer performance (Brown 2007: 117). Introducing mapping techniques and user interfaces on a pre-college level is new and necessary because they are essential to use the full and unique potential of a DMI (see Section 2.1). The essential parts of this mapping technique consist of:

1. basic math
2. basic boolean operators
3. comparison operators
4. assignment operator
5. relay switch
6. a module or system to order all this logic and math in time.

Because mapping is in fact programming, the components can be a lot more extensive, but this selection provides a basic set to connect user interface and DSP in different ways. Apart from teaching students how these mapping techniques work, it is also important to create an implicit awareness of mapping by exposing them to a diverse range of mapping solutions in performance exercises and compositions.

2.3 Teaching philosophy

Performance requires a high level of skills and automatic techniques. These are feedback loops between the sensoritory information and body gestures. Our brain receives perceptions from our ears, eyes, hands, etc., processes them in a conscious and inconscious way, and creates intentions that are embodied in body gestures which adapt and change the sound (Leman 2007: 160). This almost simultaneous cycle of perception, cognition, intention and movement is action-based, and all the separate units are related to the central act of performance. As P. R. Webster (2002: 418) observes in an article on computer-based technology and music teaching, 'constructionist thinking has been given focus in writings on school reform'. In constructionism, he continues, 'learning is seen as more effective when approached as situated in activity rather than received passively'. When teaching, live electronics is based on an action-based methodology, the theoretical knowledge can be integrated in listening and performance experiences and can foster further progress in sound imagination, experimentation and performance. Music theory and technical knowledge are tools to develop performance and general musical skills.

The above mapping techniques may seem boring or very theoretical in a music course, but if they are integrated in performance and instrument design tasks they can be fun. It is possible to play, test and hear whether the result of a mapping logic is right or wrong, and consequently, to recognize a problem and try to solve it (a method that math or programming teachers would certainly find very attractive).

The attention given to creativity in music education has grown throughout the last few decades (Hallam and Creech 2010b: 105), but the multimodal information, the lack of detailed scores and the modular nature of a DMI in live electronics (see Section 2.1) provide an extra impetus to this creativity. Creativity in music also has a clear link to aural skills, as Hallam and Creech (2010a: 92) note when they discuss creative activities in instrumental tuition.

As previously described (in Section 2.1), the number of sound parameters that can be changed and processed in electronic music are immense, and thus in this domain performance and listening go hand in hand. Developing these listening skills, as well as encouraging personal autonomy and creativity, therefore plays a central role in teaching live electronics. For example, as there are no fixed and absolute rules about the right coupling of user interface and DSP, it is very important that children have sufficient time and freedom to experiment with those techniques in order to learn more about the different factors (available equipment, physical skills, artistic demands) on which such a coupling choice depends. These experiments also help alert them to the extent that the user interface and mapping define the audio result, even when the same DSP techniques are used.

In line with the previously described teaching philosophy, tools to teach and learn live electronic music are best when they:

- stimulate the immediate perception and performance of sound
- enhance the creativity and autonomy of the user
- integrate theory and aural training in the perfomance of sound
- utilize the modularity of a DMI.

3 OPEN-SOURCE MUSIC SOFTWARE IN LIVE ELECTRONICS EDUCATION

The Abunch library tries to reach these four goals. Before describing it, we need to ask the more general question of whether open-source music software is suitable to teach and learn live electronics. In the next section, Pd (the Abunch development tool) can help answer this question.

3.1 A continuous learning environment

Anyone who wants to learn to play an instrument should have regular access to an instrument. This is a self-evident truth in instrument training. The simple but very powerful argument in favour of open-source music software for live electronics education is its accessibility, low cost and ability to run on several operating systems. These features enable schools and students to install and use this software at school and at home. In this way, they can have regular access to their Digital Musical Instrument and can start developing all the subtile automation and gestures required to perform music on an instrument. As in acoustical instrument training, the main part can be done at home, while the classroom serves to guide the continuous process. Ericsson (2006: 692) estimated the total time spent on 'deliberate practice' by music performers on acoustic instruments once they have reached the age of 20. Although the difference (8000 hours) between an expert performer and an amateur is huge, even the amateur performer cannot spend this amount of time on practising solely by playing in the classroom. He needs to have an instrument at home or another accessible place. Moreover, research stressed the importance of extra-curricular activity (Hallam 2004: 165) and non-formal learning (McQueen and Varvarigou 2010: 159) in the music learning process. Both activities are impossible without regular access to an instrument. The widespread availability of computers in households and open-source software for real-time manipulation of sound create the material conditions for a

large group of people not just to occasionally familiarize themselves with live electronic music, but to learn how to play electronic music continuously.

3.2 The combination of user software and programming language

At present, there is a wide choice of open-source software for live electronic music (Pure Data, ChucK, SuperCollider, etc.). All these programs are combinations of user software and audio programming languages. It is possible to play an off-the-shelf sound machine with these programs, and it is also possible to adapt and rebuild it by reprogramming.

This combination has some pedagogical advantages – in accordance with the teaching philosophy described earlier – especially in a graphical environment such as Pd where there is no compiler and no split between the source code and the compiled program.[7] A user can not only play with a delay effect, but he can also see how the delay effect was built. This kind of transparency helps a tutor to deal with theoretical knowledge within a performance context and corresponds very well to the action-based philosophy of live electronics courses. At the same time, this transparency also helps to abstract or transcend the software used in the classroom and to learn more about electronic music and its performance. By seeing the basic source code and learning more about fundamental techniques, it becomes easier to recognize similar procedures in other software for live electronic music.

Finally, as programming languages these aforementioned open-source music software packages have all the basic tools for learning and applying the mapping techniques. Understanding and using the modular nature of a DMI becomes feasible in a live electronics course using this kind of software.

Some pedagogical problems – especially on a pre-college level – may arise because of this combination of user software and programming language.[8]

First, a beginner can easily get lost in the massive, confusing range of possibilities and lose the motivation to learn more about electronic music.

Second, if a novice wants to find information (articles, websites, mailing lists, books) about open-source software for live electronics, it is often quite technical and requires a great deal of inside knowledge. For a beginner, some texts or mailing lists look like cryptograms. Although many patches and example files for beginners do exist in a program such as Pd, the level is often too high for pre-college teenagers, especially for those who have no background in electronic music or software programming.

Third, when I started teaching live electronics with Pd to teenagers in a music school, I noticed that there is another disadvantage to this combination: for a musician who wants to perform or compose, it takes too long before he can be creative with the instrument design. He has to know too many basic units and syntax rules before he can produce sound and start combining them.

4 ABUNCH

4.1 Content

Abunch is a collection of 60 files in two parts:

- Pd files (so-called abstractions) to perform, analyse or listen to electronic music
- information files that demonstrate or explain techniques and musical applications.

7. Pd is a visual programming language; a user can create a program (called patch or file) by manipulating program elements graphically rather than by typing a text. An object in Pd is any graphical element that performs a specific task. Apart from changing the program elements (and thus the source code), the designed patch can also be executed within Pd, and thus there is no split between the source code and the compiled program.

8. It is no coincidence that most software packages for live electronic music are programming languages. This reflects the fundamental changes in digital musical instruments and in the performance of live electronic music described in Section 2.1

The first part, the abstractions, provide a set of ready-made objects to

- record and play sound files (from hard disk and memory)
- manipulate and process sound (effects)
- generate sounds (synthesizers)
- prepare control[9] data (sequencers with different graphical interfaces)
- synchronize control data (clocks)
- analyse sound and control data (oscilloscope, spectrum analyser)
- record control data to a score
- receive data from common interfaces
- algorithmically generate control data.

These Abunch objects use techniques such as FM (Frequency Modulation) synthesis, granular synthesis and random walk algorithms. The majority of the abstractions were developed by the author, but about one-fourth is based on files by other authors.[10] All Abunch objects share a common architecture and can easily be connected with one another (and with native Pd objects) to create all kinds of custom-made live electronics.

To start off with performing, Abunch also contains a great deal of information and documented patches. Each Abunch object has a help file that explains how it works and that is accessed using the normal help procedure in Pd (right-clicking an object). Moreover, there are more than 40 example files that not only demonstrate the general application rules of Abunch objects, but also the internal structure of general audio techniques (FM synthesizer, loopstation device) and general musical 'recipes'. The latter uses guidelines and tricks to apply specific techniques that have been developed in the past 60 years by composers and performers in electronic music of different styles.

Finally, a Quick Start tutorial was developed for Abunch, as well as a mapping tutorial. The tutorial provides some simple examples for connecting user interface and sound production using the basic operators explained in Section 2.2.

Abunch is an active toolbox for experiencing and learning electronic music. Students can immediately start experimenting with computer sound by combining these objects and techniques to create their own desired sound devices. Users can listen to basic tools and techniques and actively learn more about such techniques. The open-ended architecture is based on historical examples such as the Music Logo software (1979) of Jeanne Bamberger (Holland 2000: 248; Smith 2000: 224). It encourages an experimental attitude and a critical, personal opinion. The analysing objects in the Abunch library enable the students to test and evaluate the other objects and their own built patches, and thus help them take control of their own learning.

4.2 Simplified procedures

To enable novices to produce and perform music at an early stage, the Abunch files are high-level objects, with a musical function and a graphical user interface, in contrast with low-level objects that function as basic operators to program sound. Thus, some of the previously described problems of open-source music software in Section 3.2 can be solved.

The ease of use for beginners was also obtained by simplifying installation. The installation is straightforward and only requires the Pd core version and the Abunch folder with files. The extended version of Pd (Pd-extended) is not

9. Within a real-time audio program such as Pd, audio signals are fast and continuous streams of numbers. Control data are in general slower and sporadic.

10. Authors such as Miller Puckette, Frank Barknecht and Tristan Chambers.

11. To use the Abunch files, a Pd version higher than version 0.40 (from 2006) needs to be installed.

12. Giving every Abunch object an unique argument enables storage of several instances of the same object in the presets.

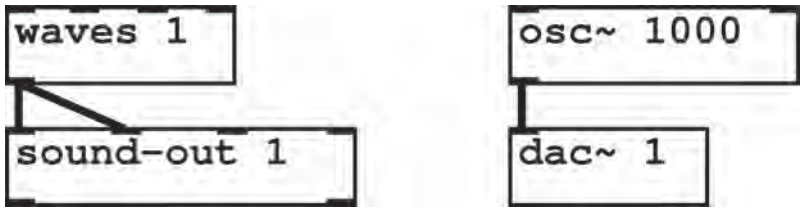

Figure 1: The different procedures in Abunch (left) and Pure Data (Pd) (right). In Abunch the argument (the number after the object name) always refers to the preset system. In Pd there are more options: '1000' refers to the frequency of the oscillator 'osc~' object and '1' to one audio hardware output of the digital-to-analog converter 'dac~'.

used, although it has many extra possibilities and functions. In Pd-extended, the extra functions can be slightly different, depending on the computer operating system. The Pd core version (called 'Vanilla') functions without difficulty in most operating systems (Linux, Mac Os X, Windows), and thus all Abunch files can be used at home.[11]

The existing procedure in Pd to create objects or connect them was simplified. To create a new object in Pd, just use the name, a ~ sign to discriminate audio from control objects, and provide an additional number or word (arguments), which refer to specific parameters and functions of that object. In Abunch, to start creating new objects, simply type the name (ignore the ~) and use a unique number as an argument. Thus, only one type of argument is used (to enable a general preset system).[12]

Once a new object is created, it is possible to start connecting objects. In Pd, objects can receive numbers, audio signals, lists or all kinds of messages for special functions. In Abunch, the connection types were reduced to numbers (for control data), audio signals and two special connection types: a clock

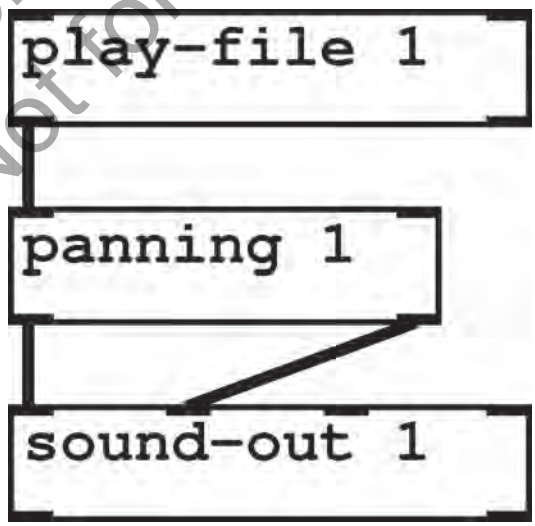

Figure 2: Three Abunch objects connected to one another. The control window of every object can be closed or opened.

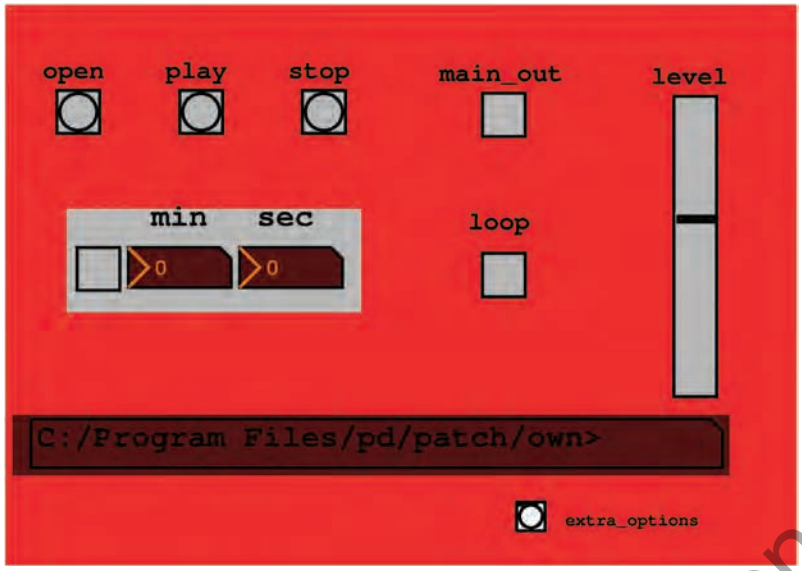

Figure 3: The control window of the 'play-file' object.

signal to synchronize time-related objects and a record connection to combine record and play objects into live recording units. Moreover, the control values are normalized to a range of 0–127. This facilitates the use of MIDI hardware to control Abunch objects and simplifies the connection between objects. Finally, the colour of the interface of the Abunch files indicates whether they are an audio object (producing a fast and continuous stream of numbers) or a control object (producing sporadic numbers).

4.3 Advanced features and integration within Pd

Abunch was launched in the beginning of 2008, and during the next 2.5 years it was tested and used by about 80 students at a pre-college music school. From October 2008 until the present, about 30 performing music students from the School of Arts in Ghent (B) also started using it in a one-semester course of live electronics. This long testing period and large user group enabled us to detect and solve many bugs. We also added extra capabilities for pedagogical purposes and student requests.

As these students became acquainted with the library, we started noticing that Abunch became too easy and restricted for some of them, so we had to find a way to combine user friendliness and more advanced features. One of the solutions was to hide the more advanced procedures and use the wireless sends and receives in Pd. Thus, the restrictions of the previously described MIDI-like procedure (in Section 4.2) to connect control objects can be superseded. Every Abunch object can print out a list of hidden send and receive names to which values within any range can be sent. Via this hidden procedure, additional parameters can be controlled and adjusted.

At a later stage, students can start using native Pd objects to add more possibilities to Abunch. One part of the example files in Abunch demonstrates how these native Pd objects can be combined with Abunch objects.

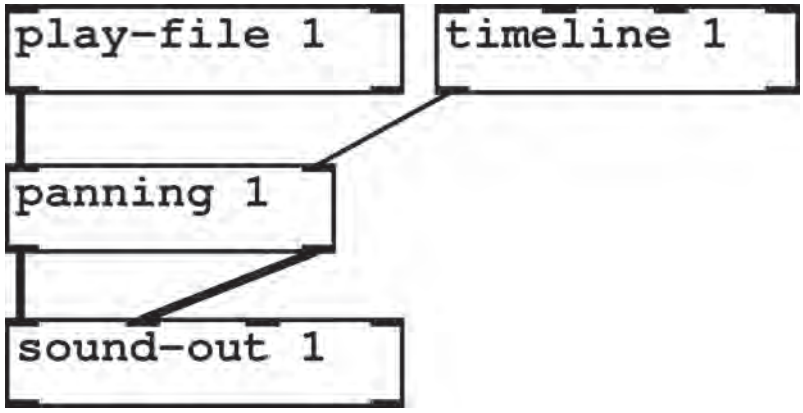

Figure 4: Normal procedure in Abunch: a sequencer object called 'timeline' controls the pan fader – invisible in this figure – within the panning object by connecting it to the right input. This input of the 'panning' object is normalized to the range 0–127, as is the output from timeline.

The procedures in Abunch are made as similar as possible to Pd procedures to aid the transition from Abunch to Pd. In general, Abunch uses a reduced number of procedures, and thus the main challenge for the transition is to learn new methods and possibilities. Two specific changes were added to Abunch (and these were mentioned in Section 4.2): the '~' is not used in the name of audio objects, and the argument of an object only refers to the general preset system.

4.4 Future plans

Abunch is still a work in progress, with room for improvement, especially because it is a solo project and the pace of development is mostly dictated by short-term educational needs.

A frequently heard criticism from students is the simple layout. Another problem is the large number of files needed to use abstractions and presets

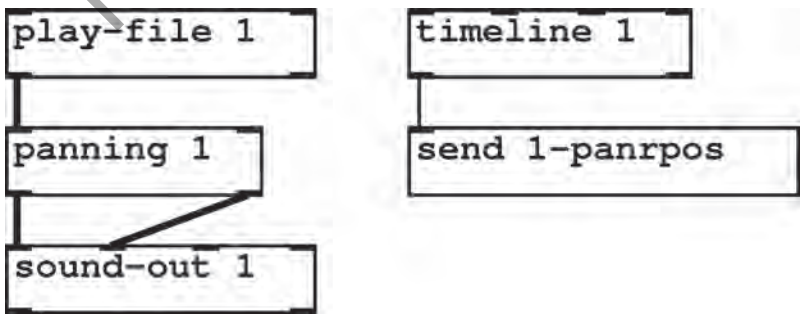

Figure 5: Advanced procedure: the range of the values in timeline can be adjusted in the extra options of this object and are sent to the send name '1-panrpos' of the pan fader in the panning object.

within a main file. There is no direct method to bundle all the files in one folder,[13] and only a workaround solution was found. This problem is pedagogically relevant because pupils perform and practice at home and often forget to copy a number of files when they return to the classroom. Finally, the problem of the score format remains unsolved. Although Abunch is intended for live electronic music, on occasion scores can be useful, e.g. to automate the playing of part of the music. The current score format in Abunch is simple and can only be edited in a text program such as Notepad, which is not very practical. In the current versions of Pd, conversion of control data to a MIDI or MusicXML format is impossible.[14]

Future versions of Abunch will include the following:

- more example files (about the musical application of techniques)
- more usage of the data structures in Pd to develop a more attractive and diverse layout
- a neat, uniform structure within each object, with information and comment in the source code
- a style guide for other developers to make new Abunch (or similar) objects
- missing essential devices from electronic music practice.[15]

5 CONCLUSION

The modular nature of a Digital Musical Instrument and the related mapping techniques are considered central parts of any live electronics course in pre-college education. Open-source music software for live electronic music is well adapted to teach these mapping techniques and its low cost and accessibility ensures the tutor and student regular access to their musical instrument, which is a prerequisite for any course in instrument training. Therefore, it is possible to use open-source music software such as Pd successfully to teach children to perform live electronic music.

A solution to the massive amount of possibilities and insufficient user friendliness in an audio programming language such as Pd is the development of a library of high-level objects such as Abunch. This library is a balanced mix of performance and theory-orientated objects with simplified ready-to-use procedures and more advanced hidden features.

REFERENCES

Australian Bureau of Statistics (2011), 'Australian social trends', http://www.abs.gov.au/ausstats/abs@.nsf/Lookup/4102.0Main+Features60Jun+2011. Accessed 23 December 2011.

Brown, A. (2007), *Computers in Music Education: Amplifying Musicality*, New York: Routledge.

Burns, A. M. (2008), *Technology Integration in the Elementary Music Classroom*, Pap/Com., Milwaukee, USA: Hal Leonard Publishing Corporation.

Eimert, H. (1957), 'What is electronic music?', in H. Eimert et al. (eds), *Die Reihe Vol. I: Electronic Music*, London: T. Presser Co., pp. 1–10.

Emmerson, S. (2007), *Living Electronic Music*, Hampshire, UK: Ashgate Publishing, Ltd.

Ericsson, K. A. (2006), 'The influence of experience and deliberate practice on the development of superior expert performance', in K. A. Ericsson

13. There is no procedure or object in the core version of Pd to know the current file name, to copy files and to create a new folder.

14. Within the core version of Pd, this is impossible. In Pd-extended solutions are possible.

15. For example, Abunch has two basic samplers in which it is possible to load one sample. Although it is possible to create several instances of this sampler simultaneously, it would also be practical and musically interesting to have one advanced sampler for loading ten or twenty samples at once. Moreover, in the two basic samplers, the pitch and duration cannot be changed independently.

et al. (eds), *The Cambridge Handbook of Expertise and Expert Performance*, Cambridge: Cambridge University Press, pp. 683–703.

Eurostat (2011), 'Households – availability of computers', http://appsso.eurostat.ec.europa.eu/nui/show.do?dataset=isoc_ci_cm_h&lang=en. Accessed 23 December 2011.

Gariépy, L. and Décarie, J. (1984), 'A system of notation for electro-acoustic music: A proposition', *Interface*, 13: 1, pp.1–74.

Hallam, S. (2004), 'How important is practicing as a predicator of learning outcomes in instrumental music?', in S. Lipscomb et al. (eds), *Proceedings of the 8th International Conference on Music Perception & Cognition*, Adelaide, Australia: Causal Productions, pp. 165–168.

Hallam, S. and Creech, A. (2010a), 'Learning to play an instrument', in S. Hallam and A. Creech (eds), *Music Education in the 21st Century in the United Kingdom: Achievements, Analysis and Aspirations*, London: Institute of Education, pp. 85–104.

—— (2010b), *Music Education in the 21st Century in the United Kingdom: Achievements, Analysis and Aspirations*, London: Institute of Education.

Holland, S. (2000), 'Artificial intelligence in music education: A critical review', in E. R. Miranda (ed), *Readings in Music and Artificial Intelligence*, Amsterdam: Harwood Academic Publishers, pp. 239–274.

Jordà, S. (2008), 'Interactivity and live computer music', in N. Collins and J. d' Escrivan (eds), *The Cambridge Companion to Electronic Music*, Cambridge: Cambridge University Press, pp. 89–106.

Karman, G. G. (2012), 'Closing the gap between sound and score in the performance of electroacoustic music', in K. Coessens and P. De Assis (eds), *Sound and Score*, Subseries Orpheus Centre in Music, Leuven: Leuven University Press.

Leman, M. (2007), *Embodied Music Cognition and Mediation Technology*, 1st ed., Cambridge, MA: The MIT Press.

McNutt, E. (2003), 'Performing electroacoustic music: A wider view of interactivity', *Organised Sound*, 8: 3 pp. 297–304.

McQueen, H. and Varvarigou, M. (2010), 'Learning through life', in S. Hallam and A. Creech (eds), *Music Education in the 21st Century in the United Kingdom: Achievements, Analysis and Aspirations*, London: Institute of Education, pp. 159–75.

Miranda, E. R. and Wanderley, M. M. (2006), *New Digital Musical Instruments: Control And Interaction Beyond the Keyboard*, Pap/Com., Middleton, Wisconsin: A-R Editions.

Office for National Statistics (2011), 'Ownership of consumer durables increases into 2010', http://www.ons.gov.uk/ons/rel/family-spending/family-spending/family-spending-2011-edition/sum-consumer-durables-nugget.html. Accessed 23 December 2011.

Oore, S. (2005), 'Learning advanced skills on new instruments', in *Proceedings of the 2005 Conference on New Interfaces for Musical Expression. NIME 2005*, Singapore: National University of Singapore, pp. 60–64, http://dl.acm.org/citation.cfm?id=1085939.1085958 Accessed 12 December 2011.

Pestova, X. (2009), 'Models of interaction: Performance strategies in works for piano and live electronics', *Journal of Music, Technology and Education*, 2&3: 23, pp. 113–26.

Reese, S., McCord, K. and Walls, Kimberly (eds) (2001), *Strategies for Teaching: Technology*, Reston, VA, USA: MENC, the National Association for Music Education.

Rudolph, T. E., Richmond, F., Mash, D., Webster, P., Bauer, W.I. and Walls K. (2005), *Technology Strategies for Music Education,* 2nd ed., F. Richmond, ed., Wyncote, PA, USA: Hal Leonard Publishing Corporation.

Savage, J. (2009), 'Hand2Hand and Dot2Dot: Developing instruments for the music classroom', *Journal of Music, Technology and Education*, 2: 23, pp. 141–57.

Smith, B. (2000), 'Artificial intelligence and music education', in E. R. Miranda (ed), *Readings in Music and Artificial Intelligence*, Amsterdam: Harwood Academic Publishers, pp. 221–238.

Statbel (2011), 'ICT-indicatoren bij huishoudens en individuen (2005–2010)', http://statbel.fgov.be/nl/modules/publications/statistiques/arbeidsmarkt_levensomstandigheden/ict_indicatoren_bij_huishoudens_individuen.jsp. Accessed 23 December 2011.

Webster, P. R. (2002), 'Computer-based technology and music teaching and learning', in R. Colwell and C. Richardson (eds), *The New Handbook of Research on Music Teaching and Learning*, New York: Oxford University Press, pp. 416–39.

SUGGESTED CITATION

Roels, H. (2012), 'Abunch, a tool to teach live electronics in pre-college music education', *Journal of Music, Technology & Education* 5: 2, pp. 181–193, doi: 10.1386/jmte.5.2.181_1

CONTRIBUTOR DETAILS

Hans Roels (1971) is a Ph.D. researcher in the School of Arts, University College Ghent, Belgium (www.schoolofarts.be) where he teaches live electronic music. Since 2010, he also works as a researcher in the Orpheus Research Centre in Music (ORCiM)(www.orpheusinstituut.be). Hans Roels studied piano and composition, and during the fifteen years that he was active as a professional composer his works were played in several European countries. Between 2001 and 2008, he was responsible for the concert programming in the Logos Foundation, a centre for experimental audio arts (www.logosfoundation.org). Between 1995 and 2010, Hans Roels was a music teacher (piano, harmony, electronic music) at the music academy of Deinze (B), a music school for children and amateur musicians. More info: www.hansroels.be

Contact: Royal Conservatory, School of Arts, University College Ghent, Hoogpoort 64, B-9000 Ghent, Belg
E-mail: hans.roels@hogent.be