

MAMIC goes live: a music programming system for non-specialist delivery

Item Type	Chapter in book
Authors	Dalgleish, Mat;Payne, Christopher
Citation	Dalgleish, M. and Payne, C. (2019) MAMIC goes live: a music programming system for non-specialist delivery, in Hepworth-Sawyer, R., Hodgson, J., Paterson, J. and Toulson, R. (Eds.) Innovation In Music: performance, production, technology and business. London: CRC Press
Publisher	CRC Press
Download date	2026-03-07 18:58:40
License	https://creativecommons.org/licenses/by-nc-nd/4.0/
Link to Item	http://hdl.handle.net/2436/622324

Abstract

The computing curriculum in England has shifted from “software training” to a model where children learn to code as a way of understanding underlying principles. This has created challenges for primary school teaching practitioners, many of whom require upskilling. The Music And Math In Collaboration (MAMIC) project addresses these issues via a custom library for the Pure Data (Pd) visual programming environment that interconnects key musical, mathematical and coding concepts within a unified environment that is able to be delivered by non-specialist teachers. Initial findings from deployment “in the wild” (i.e. in situ) are presented and future work is discussed.

Running Head Right-hand: MAMIC Goes Live

Running Head Left-hand: Mat Dalglish and Chris Payne

MAMIC Goes Live

A Music Programming System for Non-specialist Delivery

Mat Dalglish and Chris Payne

Towards Computational Thinking in Schools

The English national curriculum sets out the subjects taught, the contents of these subjects, targets for achievement, and standards for each subject, in all local authority–maintained primary and secondary schools nationwide. The last six years have seen controversial changes to some subjects, most notably Mathematics and Computing. For instance, the new Mathematics curriculum requires first-year primary students (ages 5–6) to handle and manipulate large numbers. It also requires the earlier introduction of more advanced topics such as fractions and algebra, before they are revisited in more detail in later years ([Dominiczak, 2013](#)). The change of ethos in relation to Computing is arguably more radical still: the new curriculum suggests that students should not only become adept users of software (i.e. the Information and Communications Technology [ICT] model) but also understand the fundamental principles that underpin the creation of software artifacts ([Andrews, 2014](#); [Curtis, 2013](#)). This ultimately means that students are not only taught how to code, but also encouraged to develop more abstract skills that relate to Computational Thinking.

Following an initial but extensive definition of Computational Thinking by [Wing \(2006\)](#), [Selby and Woollard \(2014\)](#) describe how the term has been contested and a multitude of different definitions proposed. However, there is broad agreement that Computational Thinking relies on multiple stages to solve complex problems (Selby and Woollard, 2014). These stages can be broadly described as follows ([Kemp, 2014](#)):

MAMIC Goes Live

- Decomposition—breaking down problems into manageable components
- Pattern recognition—analysis of identical or similar states in a system so that they can be appropriately categorized, in order to permit the repeated use of a concept
- Abstraction—streamlining a proposed system for reasons of efficiency and stability, by applying only essential concepts
- Algorithm design—designing systems to solve a given problem

For [Dehnadi and colleagues \(2009\)](#), the ability to think computationally is a useful indicator of practical performance. More specifically, they found that those students who are able to develop and consistently apply a mental model of programming were more than twice as likely to pass a practical programming test. For [Berry \(2013\)](#), however, the benefits of Computational Thinking extend beyond the activity of programming or even the discipline of Computing, to provide a kind of toolkit that can be applied to problems in diverse scenarios. To this end, he states that: “Computational thinking provides insights into many areas of the curriculum, and influences work at the cutting edge of a wide range of disciplines” ([Berry, 2013](#)). This is notably considered a boon to the modern economy, as well as pedagogically expedient.

Teachers and Their Discontents

Despite the importance placed on the shift from the old ICT model to the current computing curriculum, [John \(2013\)](#) notes that the Graduate Teacher Training Registry (GTTR) saw a 41% decline in graduates training to become ICT secondary school specialists in 2012–13. A 50% decline in the number of computer science teaching applicants was also reported in the same academic year ([John, 2013](#)). These issues lead Connell, chair of the Association for Information Technology in Teacher Education (ITTE), to state that:

MAMIC Goes Live

[The ITTE is] less confident than the Teaching Agency (TA) that we will have the necessary workforce in place to effectively deliver the new Computing Curriculum for 2014. We believe it will take longer—a view reflected by the BCS itself in its submission to the consultation on the new curriculum.

(Connell, in [John, 2013](#))

There are also concerns from the perspective of existing teachers. [Andrews \(2014\)](#) states that teachers and institutions are skeptical about the ongoing implementation of the new computing curriculum. The broad consensus among many educators is that there is a lack of skills relating to programming specifically within the teacher population. For example, in July 2014, a YouGov survey commissioned by the innovation charity Nesta and Tes (previously the Times Education Supplement) found that 60% of ICT teachers did not feel confident about the new curriculum delivery that was implemented in September 2014. Moreover, over half the surveyed teachers admitted to not seeking any advice on the subject before the start of the academic year. Overall, 67% of teachers surveyed stated that despite the government-funded initiatives, they still did not feel they had adequate support and guidance to aid their subject knowledge and support lesson planning activities ([John, 2014](#)).

In an attempt to address preconceptions (including those of teachers), the Year of Code project was established in 2014, with government backing ([Year of Code, 2015](#)). Although Year of Code has been praised by some ([Cellan-Jones, 2014](#)), teaching groups and academics remained concerned about a lack of necessary skills, prompting a further £3.5 million government training scheme, Computing at School (CAS), relating to computational thinking. The CAS initiative aims to support the new ICT curriculum with training programs and subject resources, as well as instigating reform of awarding body criteria ([Computing at School, 2015](#)). In particular, CAS advocates the creation of self-sufficient “Master Teachers” who are trained to deliver professional development sessions to teaching staff across a range of institutions ([Humphreys et al., 2014](#), p. 4). However, while it was anticipated that 400 Master

Teachers would be created, [Dickens \(2016\)](#), p. 1) comments that it is unclear how many are currently in post.

Locating Music Technology

While increased importance has been placed on technology (and computer technology in particular) elsewhere in the National Curriculum, the state of music technology is less clear. As far back as 1997, [Innes \(1997\)](#) noted that students are reliant on “forward-looking teachers” to implement technological approaches within music. In the two decades since, music technology at school level has continued to be available for study only indirectly, through the inclusion of some of its facets in related subjects, or informally ([Gehlhaar, 2002](#)). Indeed, the most recent National Plan for Music Education (NPME) document ([Department for Education, 2011](#)) makes it clear that music technology is still considered only as an addition to the music curriculum rather than a subject in its own right. Interestingly, the report includes a number of examples of “appropriate” use of technology within music ([Department for Education, 2011](#)). These include:

- sound recordings to aid music listening
- sequencers, loopers, samplers and effects processes for the purpose of musical composition and performance
- online resources and smartphone apps as ways to assist instrumental practice and learning music theory, respectively
- online collaboration
- digital keyboards as a cost-effective replacement for acoustic pianos
- digital audio workstations as a cost-effective replacement for the recording studio

MAMIC Goes Live

If hardly novel—many of these practices are already established outside of the school domain—the NPME report still asserts that technology in music must be used wisely and not distract from musical or lesson content. At the same time, it also notes that some teachers make good use of music technology, but it is underused by others. Closely related, the [Paul Hamlyn Foundation \(2014\)](#) comments that: “Music technology is not yet sufficiently integrated into school-based music, and many teachers do not capitalize on pupils’ confidence and facility with technology”.

If the 2011 NPME report suggests that “further work should be undertaken to develop a national plan for the use of technology in the delivery of Music Education—and to ensure that the workforce is up-to-date with latest developments”, efforts to date are considered insufficient by some. For instance, as part of a public consultation in February 2013 led by the Secretary of State for Education and intended to implement curriculum reforms by September 2014, respondents commented that the role of Music Technology required further definition ([Department for Education, 2013](#)).

One particular debate centers around whether Music Technology should be a stand-alone subject. Certainly, a case can be made for the creation of a stand-alone Music Technology subject at school level. For instance, music technology degrees have come to feature prominently in Higher Education (Boehm, 2005), and Music Technology awards are firmly established in Further Education. It could therefore be argued that a specialized Music Technology award at school level might better prepare students for progression onto music technology routes in Further Education and Higher Education. It may also increase recognition and funding for the area, and increased study time might enable students to explore the discipline in greater detail. Nevertheless, a case can also be made for an alternative approach: the spread of music technology into other subjects. For instance, from the perspective of employment, traditional music technology career paths such as studio engineer and producer have suffered significantly with the demise of many large recording studios ([Levshon, 2009](#)) and the decline of recorded music sales. At the same time, new opportunities have emerged elsewhere. These have largely centered around the intersection of music technology with other disciplines, sometimes in combinations that would have seemed unlikely in even the relatively recent past. For

MAMIC Goes Live

example, while [Thorley \(2005\)](#) comments that “there is no such job as a ‘music technologist’”, music technology specialists make significant contributions to fields such as:

- film and television
- theatre
- video games
- virtual environments and augmented reality
- architectural and archaeological acoustics
- product and industrial design
- Human-Computer Interaction (HCI)
- signal processing
- software development

These intersections are closely linked to the flexibility of the computer, as well as its ubiquity. The ability of the home computer to approximate what was previously only achievable using a collection of specialized and expensive studio equipment is well documented ([Leyshon, 2009](#)). However, arguably more impactful is the ability of the computer to act as a kind of proxy that is able to leverage skills previously confined to one discipline into new areas—a means of traversing previously largely impermeable disciplinary boundaries. For instance, open-ended programming environments such as Max, Pure Data (Pd) and SuperCollider enable translation between senses: video can become sound, sounds can become rendered 3D graphics, and—aided by low-cost microcontrollers such as the Arduino—either domain can be turned into control signals for a host of physical actuators. Equally important is that these environments recognize that potential users often come from arts backgrounds rather than computer science and make programming concepts accessible to these groups.

As the tendrils of music technology spread into and entangle with other areas, and with [Sawyer et al. \(2013\)](#) stating that music act as a vehicle to expose students to a science, technology, engineering and math (STEM) curriculum, it is useful to consider the notion of transfer learning. Transfer learning relates to the notion that, unconsciously or through concerted reflection, skills developed in one area can transfer to other areas that involve similar processes ([Hallam, 2010](#)). For Fleishman, these transfers are an everyday—but vital—occurrence:

Transfer of learning . . . is pervasive in everyday life, in the developing child and adult. Transfer takes place whenever our existing knowledge, abilities and skills affect the learning or performance of new tasks . . . transfer of learning is seen as fundamental to all learning.

(Fleishman, in [Cornier and Hagman, 1987](#), p. xi)

[Hallam \(2010\)](#), however, also notes that transfers are not all equal, or equally likely. “Near” transfers are where concepts and performances are closely related. “Far” transfers are between poorly matched contexts and performances. Hallam argues that near transfers are stronger and more likely to occur.

Music and Mathematics

If music and computing have become closely aligned, it has long been identified that mathematics has a close affinity with music. For instance, as early as 500 BCE, the Ionian Greek philosopher Pythagoras observed the integral relationships between frequencies of musical tones in a consonant interval ([Wright, 2009](#)), likely leading to the development of the Greek musical scales ([Hawkins, 2012](#)).

If musical rudiments such as pitch and rhythm can be described as fundamentally mathematical, many 20th-century musical works are based on specialized contemporary mathematical concepts. For instance, the musical material of Iannis Xenakis’s *Pithoprakta* (1955–56) is rooted in the statistical mechanics of gases, and its title refers to Bernoulli’s law of large numbers: namely that, as the number

of occurrences of a chance event increases, the more the average outcome approaches a determinate end ([Antonopoulos, 2011](#)). Other examples include: (a) the use of the Monte Carlo method of random number generation to make organizational decisions in Hiller and Isaacson's *Illiad Suite* ([Sandred et al., 2009](#)); (b) the use of multidimensional crystal algorithms to generate harmonic structures in the work of James [Tenney \(2008\)](#); (c) the use of self-similar structures, particularly melodies, by [Tom Johnson \(2006\)](#); and (d) the use of a Game of Life algorithm to generate the intervals of a triad based on the locations of active cells in Eduardo Miranda's *CAMUS* ([Burraston et al., 2004](#)).

As interest in the relationship between mathematics and music continues ([Wright, 2009](#); Loy, 2006; [Tymoczko, 2011](#)), there is also some empirical evidence that linking the two disciplines may be of pedagogical benefit for school-age children. For instance, a study by [An et al. \(2013\)](#) involved two student teachers receiving training in relation to the integration of music and mathematics from experts, before incorporating music activities into their usual mathematics classes with 46 first- and third-grade schoolchildren from a range of backgrounds. Activities included composition and performance, delivered as ten 45-minute sessions in total over a period of five weeks. Before the classes began, a series of five Model-Strategy-Application (MSA) tests were administered to every participant, and a further five MSA tests were administered after each interdisciplinary session. [An et al. \(2013\)](#) found that the performance of both first- and third-grade students in all three mathematical areas assessed by the MSA test showed statistically significant improvements after the introduction of the interdisciplinary curriculum.

MAMIC: Music And Mathematics In Collaboration

The 2011 National Curriculum in England has changed how mathematics and, particularly, computing are taught in schools. However, despite training schemes backed by the government, teachers continue to feel that they are underprepared for the changes. At the same time, the role and position of music technology, itself increasingly computing-centric, remains poorly defined. Responding to these issues,

MAMIC Goes Live

Chris Payne's Music And Math In Collaboration (MAMIC) project is intended to be used by children aged 9–11 years and aims to:

- leverage musical process relating to composition and performance to introduce coding (and visual programming specifically) in an accessible and engaging way
- use musical process to vibrantly introduce mathematical concepts
- reinforce existing mathematical knowledge by making abstract concepts audible and/or visible, and able to be modified in a hands-on way
- facilitate and support “in the wild” delivery by non-specialist teachers
- run on a wide variety of computers, including on the older and budget hardware found in many schools, without requiring permanent installation

The MAMIC project ([Payne, 2018](#)) consists of three main elements:

- a library (i.e. extension) for the Pd visual programming environment
- encapsulation of the Pd/MAMIC library combination in a portable, Linux-based live image able to run from a USB pen drive
- tutorial materials intended to aid and support delivery by non-specialist teachers. These consist of a manual, help patches, and step-by-step guides for teachers and students in video and written formats.

Pd ([Puckette, 2017](#)) is used as the overarching programming environment as a free and Open Source alternative to Max ([Cycling '74, 2017](#)). A visual programming language was chosen in order to minimize the gap between conceptual model and implementational representation, particularly in relation to data flow. Open Source is seen as a boon to enabling reuse or further development by others, and the

MAMIC Goes Live

tutorial materials are also available under a [Creative Commons \(2017\)](#) Attribution Non-Commercial license.

The live Linux image is based on the [Ubuntu Studio \(2017\)](#) distribution, packaged so that non-specialist teachers can boot the system from the pen drive without the need for hard disk installation. It is also designed to be used on the variety of computer systems that can be found in the education system and includes detailed CPU usage-reduction strategies to enable smooth real-time operation on older and budget (i.e. lower specification) hardware. The image also includes shortcuts to SimpleScreenRecorder so that desktop movements and actions can be recorded to various video formats, and VLC Media Player for the use of video playback (e.g. to watch video tutorials or self-capture). Lastly, a dedicated Student Work folder enables students to easily save their work to unused space on the pen drive.

The MAMIC library consists of 75 abstractions that each perform a different compositional or performative task or subtask. The abstractions are categorized and grouped by broad function, leading to the following hierarchy of layers ([Figure 17.1](#)):

- External Input
- Internal Sequencing
- Composition
- Sound Generation
- Sound Output

The External Input layer provides a number of abstractions aimed at capturing input from external devices and producing musically useful control data. Controllers currently supported include MIDI and computer keyboards, gamepads and webcams. Multiple input devices can be used simultaneously and external input combined with the internal sequencer.

[Insert 15031-2542-P3-017-Figure-001 Here]

Figure 17.1 An Overview of the MAMIC Library

The Internal Sequencing layer is based around a prebuilt sequencer called Conductor. The Conductor abstraction utilizes an array of 16 steps (colored orange), each able to store a user-determined sequence number between 1 and 16. Activating the Play button causes Conductor to cycle through the steps at the specified tempo (shown in the tempo display). As Conductor arrives at each step, the sequence number stored at that step is sent to its output outlet. This sequence number can then be used to trigger aspects in the composition layer such as scales, chords or one-shot audio samples.

The Composition layer operates primarily at the note level. MAMIC features prebuilt major and minor-scale objects. The Major-scale and Minor-scale abstractions contain formulae for whole and half steps. Students then select the tonic and the scale degree to be played. MAMIC also enables students to create diatonic chords (diatonic harmony) in two different ways via the Majorchords and Makechord abstractions.

Majorchords holds and creates all the diatonic chords for the selected major scale. Users can then send sequence numbers (1–8) to select the chord to be played. Roman numeral representations (as commonly used in music theory) are also displayed. Makechord displays three columns of numbers that enable a three-note chord to be constructed using the notes of a major or minor scale. When a sequence number (1–16) is sent to Makechord, the programmed chord will then play. Hidden interconnections between objects ensure that Makechord automatically “knows” the scale being used.

The Sound Generation layer contains a modest pallet of simple sound-generation options. These include a number of simplified synthesizers, plus one-shot and looping sample players. While it is considered vital that novice users can very quickly produce basic sounds, some flexibility is provided for more experienced users.

The Sound Output layer contains a simple abstraction that outputs stereo audio.

The more complex or multifunctional abstractions in the MAMIC library implement a system of tiered object granularity, as proposed by [Bukvic et al. \(2012\)](#). This is termed Differentiated Abstraction and means that abstractions are made up of multiple tiers that gradually increase in complexity. Three tiers are implemented:

- Tier 1: involves simple interactions and/or object functionality is automated
- Tier 2: requires a greater degree of interaction and/or interaction across the disciplinary boundaries of music, mathematics and coding
- Tier 3: involves more complex or sophisticated mental operations such as the creation of textual code to set real-time functions or operations

Examples of differentiated levels of abstraction as featured in an early version of the Major-scale abstraction are shown in [Figure 17.2](#).

[Insert 15031-2542-P3-017-Figure-002 Here]

Figure 17.2 An Example of Differentiated Abstraction Levels in the MAMIC System

Higher tiers can be hidden to simplify operation for younger or novice users, or revealed to provide greater freedom or to introduce more sophisticated concepts.

Testing

Testing of the MAMIC project has been iterative and multi-staged. First, postgraduate music technology students provided expert review of the library at a preliminary stage in its development. All had two or

MAMIC Goes Live

more years of experience with music programming languages and were therefore well placed to provide comment.

After two cycles of this test-and-improve process, a subsequent stage shifted the project out of the lab and into “the wild” (Rogers, 2011). A key premise of in-the-wild studies is that researchers are able to evaluate new technologies as they are really used. In particular, they acknowledge that experimental context can influence results and try to minimize this effect by operating in the participants’ usual or “natural” environment (Brown et al., 2011).

To date, one in-the-wild study has been carried out over a period of ten weeks in a Warwickshire (UK) school. A second in-the-wild study is currently underway in another school. To better understand if the MAMIC project can be delivered by non-specialists, the first study was delivered by two student teachers: both held music-related degrees (in production and composition, respectively), but neither had programmed before or studied either mathematics or computing beyond age 16. To introduce them to the project, both student teachers completed an initial 90-minute training session, before a period of self-directed learning and familiarization with query support. The student teachers then used the MAMIC materials to plan and deliver five 2-hour sessions to a class of ten Year 5 children (9–10 years old) over a period of seven weeks. Participants were of mixed ability and from diverse backgrounds. Sessions were playful and open-ended, with participants devising and subsequently exploring personal interests and goals. The sessions took place in the participants’ usual classroom environment and, for reasons of manageability, focused on a subset of abstractions that encapsulate the main aspects of the MAMIC library.

Throughout the sessions, Unmoderated Remote Usability Testing (URUT) techniques (Barnum, 2010, pp. 44–45) informed the use of automated video screen capture to document participant activities and outcomes. This focused on how participants manipulated and connected the MAMIC abstractions to create generative music (i.e. the patches created by students), but also captured any student teacher

MAMIC Goes Live

interactions or interventions inside the MAMIC environment. The video screen capture files were retrieved from the USB sticks at the end of every session and archived for later analysis.

Alongside their delivery, the student teachers kept diaries that summarized their sessions. These diary entries provided a narrative outline of each session, together with comments around any positive aspects or points for improvement.

At the end of the seven weeks of delivery, the trainee teachers also evaluated the MAMIC library according to the Heuristic Evaluation model developed by [Nielson \(1995\)](#). The heuristics are general principles that are intended to describe typical properties of usable interfaces. The model consists of the following principles:

- Visibility of system status
- Match between the system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition rather than recall
- Flexibility and ease of use
- Aesthetic and minimalist design
- Help users recognize, diagnose and recover from errors
- Help and documentation

For convenience, these heuristics were distilled into an online form with ten questions in total (one question per heuristic). Clarity statements were provided for each question to ensure understanding of its relevance to the project. Each question asked the student teachers to provide a quantitative score (1–5) for the heuristic and then provide rationale for their choice. A final section provided additional space to comment on the quality and appropriateness of the MAMIC documentation and broader experiences of deployment and operation.

Initial Findings and Discussion

Expert review from the postgraduate students tended to focus on technical and experiential aspects. It primarily helped to improve library functionality and usability through interface design and computational optimization. In the latter instance, students informed the reorganization of some abstractions into subpatches so as to enable the extensive use of localized DSP management. These adjustments appreciably aided the ability of the MAMIC library to run effectively on the aging and lower-specification hardware found in the schools.

The diaries of the student teachers in the first in-the-wild study suggest that, by the end of the initial two-hour session, participants had comprehended how to connect and set the abstractions so as to achieve basic functionality. The second session saw participants build their own patches (i.e. collections of interconnected abstractions). This suggests that they had not only become familiar with how to initialize MAMIC abstractions and change their parameters, but also started to conceive and implement their own algorithms. The student teachers adopted a rotation system reminiscent of *Cadavre exquis* ([Kochhar-Lingren et al., 2009](#)), so that each participant could experience and contribute to a variety of different algorithms. The diary entry notes that by the end of the session, “most” participants demonstrated confidence in their use of the MAMIC library and how they tackled peer creations. From the third session onwards, participants started to connect and explore the use of hardware controllers such as MIDI keyboards and video game controllers. Participants also developed

MAMIC Goes Live

more fully realized and individualized patches, and there was evidence of peer-to-peer help and problem solving. Interventions by the student teachers involved details of implementation rather than core concepts. They include a reminder of the keyboard shortcut to switch between edit and performance modes in the Pd environment, and how to swap the games controller abstraction between its sampler and sequencer modes. Overall, the student teacher diaries consistently noted that they were pleased with the progress of participants but made relatively little comment about their own experiences.

The Heuristic Evaluations completed by the student teachers at the end of the seven-week delivery period reveal that, despite differences in their backgrounds, both student teachers had largely positive experiences with the MAMIC system overall, with the production graduate rating most aspects of the system more highly than the composition graduate.

In relation to the “match between the system and the real world” and “recognition rather than recall” aspects, the student teachers commented that the MAMIC abstractions are well named and represented; the names of most abstractions and their graphical representation on screen appear to convey their particular functions to the intended users. The exceptions are the more abstract objects. For example, the student teachers identified that the “volume”, “load audio sample” and “trigger buttons” abstractions are isolated instances that could be difficult to work with without the use of additional labels. Other aspects relating to recognition rather than recall include quickly understanding the flow from one object to the next. For instance, the student teachers also noted that “students needed little guidance once shown the basics with regards to how to create and link patches”.

Responses to the “error prevention” and “help users recognize, diagnose and recover from errors” heuristics are more suggestive of areas for improvement. For instance, the composition graduate reported more system errors than the production graduate and noted that on occasion the “easiest option” was to close the MAMIC system and reload it. However, it was also noted that this was costly in

MAMIC Goes Live

terms of time and is therefore a problematic solution. Additionally, some errors did not produce visible error reports, thereby limiting opportunities for diagnosis and recovery.

In terms of the “aesthetic and minimalist design” heuristic, the student teachers commented that the construction and presentation of the abstractions was largely successful in hiding their inner complexity from participants. However, some participants did occasionally stumble into this interior. This could result in confusion and also caused some issues with source code manipulation (and is therefore also relevant to “error prevention”).

“User control and freedom” and “flexibility and ease of use” are arguably more difficult concepts to evaluate. On the one hand, the MAMIC library constrains possibilities compared to the standard, unadorned Pd environment. On the other hand, the MAMIC library still presents an almost inexhaustible number of possible musical topologies. This “guided freedom” is considered a positive aspect by the student teachers, but it is also noted that more guidance or further training would be needed to fully harness these possibilities.

In the additional comments section of the form, both student teachers mentioned the help materials supplied on the MAMIC live USB image: in particular, they found the resources to be very useful as a foundation for their self-study.

The video capture material appears to support many of the comments made by the student teachers. For example, video screen capture from the first session shows that the participants had already discovered how to control the functions of MAMIC abstractions and how to successfully interconnect them. An example of this can be seen in [Figure 17.3](#). However, when abstractions did not act as expected, the MAMIC system was not always able to respond. As a result, the participants or student teacher sometimes had to find the cause of faults on their own. Video capture also provided evidence of audio problems at a system level. Thus, after discussion with the student teachers, shell scripts were used to automate JACK audio server initialization (i.e. how and when the server loads).

[Insert 15031-2542-P3-017-Figure-003 Here]

Figure 17.3 A Simple But Functional Example Patch Created by a Participant

While it is more difficult to ascertain how quickly and fully the students developed understanding of the underlying concepts, there is evidence that they have been internalized, at least in some cases. Most notably, some compositions produced by participants do not simply repeat the applications covered in class, but instead use ideas introduced in class as the basis for improvisation and synthesis. For instance, the participant example in [Figure 17.4](#) combines two different means of input and adds a simple preset manager: either a MIDI keyboard or the inbuilt sequencer can be used for input, and sequencer patterns can be stored and recalled. There is also possible evidence of the participant experimenting with webcam control (seemingly discarded) towards the top-left corner of the screen.

[Insert 15031-2542-P3-017-Figure-004 Here]

Figure 17.4 A More Sophisticated Patch That Combines Two Different Forms of Input

Conclusion and Future Work

Findings from the first in-the-wild study have been promising in terms of how readily users have taken to the MAMIC system and the positive overall experiences of delivery. Issues arising have been primarily of a technical nature and subsequently resolved. A second in-the-wild study is currently underway in another primary school, with delivery over a 14-week period by permanent rather than trainee teachers. It is of particular interest to see how the system fares when delivered by teachers without a musical background (i.e. when the diversity of delivery increases). Indeed, the involvement of participating teachers is considered vital going forward. For instance, teachers at the second school have already asked for the system to be trialed for use with children of reception age (5–6 years old), and for dance mats to be implemented as an additional means of input. The latter takes the MAMIC project towards

what [Holland et al. \(2011\)](#) call “whole body interaction in abstract domains”. This may extend the project in at least two ways. New matches between physical action and system output may be developed. These could open the system up to new participants (e.g. anyone not able to use current input devices) or help current users to develop new understanding. Additionally, by encouraging users to be physically active, whole-body interaction also has the potential to benefit user health and fitness.

Longer-term work involves deeper exploration of how to situate the MAMIC project relative to the National Curriculum in England. This requires engagement at a variety of scales, from governmental policy, to still largely unconnected subjects, to individual lesson plans.

References

[An, S., Capraro, M. M. and Tillman, D. \(2013\)](#). Elementary teachers integrate music activities into regular mathematics lessons: Effects on students’ mathematical abilities. *Journal of Learning Through the Arts*, 9(1), pp. 1–21.

[Andrews, S. \(2014\)](#). *What’s changing in the computing curriculum?* [online]. [Accessed 14 Dec. 2017]. Available at: www.pcpro.co.uk/features/389875/whats-changing-in-the-computing-curriculum.

[Antonopoulos, A. \(2011\)](#). Pithoprakta: The historical measures 52–59: New evidence in glissando speed formalization. In: *Xenakis international symposium*. London: Southbank Centre, pp. 1–23.

[Barnum, C. \(2010\)](#). *Usability testing essentials: Ready, set . . . test!* Burlington, MA: Morgan Kaufmann.

[Berry, M. \(2013\)](#). *Computing in the national curriculum: A guide for primary teachers* [online]. Available at: www.computingschool.org.uk/data/uploads/CASPrimaryComputing.pdf [Accessed 14 Mar. 2015].

[Boehm, C. \(2005\)](#). The thing about the quotes: “Music Technology” degrees in Britain. In: *International Computer Music Conference (ICMC)*. New Orleans: International Computer Music Association, pp. 682–687.

MAMIC Goes Live

- Brown, B., Reeves, S. and Sherwood, S. (2011). Into the wild: Challenges and opportunities for field trial methods. In: *ACM CHI 2011 conference on human factors in computing systems*. Vancouver: ACM Press, pp. 1657–1666.
- Bukvic, I., Baum, L., Layman, B. and Woodard, K. (2012). Granular learning objects for instrument design and collaborative performance in k-12 education. In: *International conference on New Interfaces for Musical Expression (NIME)*. Ann Arbor, MI, pp. 344–346.
- Burraston, D., Edmonds, E., Livingstone, D. and Miranda, E. R. (2004). Cellular automata in MIDI based computer music. In: *International computer music conference*. Miami: International Computer Music Association, pp. 71–78.
- Cellan-Jones, R. (2014). *Year of code—PR Fiasco or vital mission?* [online]. Available at: www.bbc.co.uk/news/technology-26150717 [Accessed 14 Dec. 2017].
- Computing at School. (2015). *About us* [online]. Available at: www.computingatschool.org.uk/index.php?id=about-us [Accessed 21 Dec. 2017].
- Cormier, S. M. and Hagman, J. D. (eds.). (1987). *Transfer of learning: Contemporary research and applications*. London: Academic Press.
- Creative Commons. (2017). *Attribution-noncommercial 4.0 international (CC BY-NC 4.0)* [online]. Available at: <https://creativecommons.org/licenses/by-nc/4.0/> [Accessed 2 Jan. 2018].
- Curtis, S. (2013). *Teaching our children to code: A quiet revolution* [online]. Available at: www.telegraph.co.uk/technology/news/10410036/Teaching-our-children-to-code-a-quiet-revolution.html [Accessed 2 Jan. 2018].
- Cycling '74. (2017). *Max* [software]. Available at: <https://cycling74.com/products/max/> [Accessed 2 Jan. 2018].

MAMIC Goes Live

Dehnadi, S., Bornat, R. and Adams, R. (2009). Meta-analysis of the effect of consistency on success in early learning of programming. In: *Psychology Programming Interested Group (PPIG) annual workshop*. The Open University, Milton Keynes. Available at: www.ppig.org/papers/21st-dehnadi.pdf [Accessed 11 Jan. 2018].

Department for Education. (2011). *The importance of music: A national plan for music education* [online]. Available at: www.gov.uk/government/uploads/system/uploads/attachment_data/file/180973/DFE-00086-2011.pdf [Accessed 16 Dec. 2017].

Department for Education. (2013). *Reform of the national curriculum in England Report of the consultation conducted February-April 2013* [online]. Available at: www.education.gov.uk/consultations/downloadableDocs/NC%20in%20England%20consultation%20report%20-%20FINAL%20-%20Accessible.pdf [Accessed 16 Dec. 2017].

Dickens, J. (2016). *Government spends £3m in scramble to get 400 “master” computing teachers* [online]. Available at: <https://schoolsweek.co.uk/3m-on-and-where-are-all-the-master-teachers/> [Accessed 16 Dec. 2017].

Dominiczak, P. (2013). *Michael Gove: New curriculum will allow my children to compete with the very best* [online]. Available at: www.telegraph.co.uk/education/educationnews/10166020/Michael-Gove-new-curriculum-will-allow-my-children-to-compete-with-the-very-best.html [Accessed 14 Dec. 2017].

Gehlhaar, R. (2002). Music technology and sound art/interactivity: Reality music/virtual music. In *25th national conference of the Musicological Society of Australia (MSA)*. Newcastle, NSW. Available at: www.gehlhaar.org/x/doc/musictechnologymrsa.doc [Accessed 10 Jan. 2018].

Hallam, S. (2010). The power of music: Its impact on the intellectual, social and personal development of children and young people. *International Journal of Music Education*, 28(3), pp. 269–289.

MAMIC Goes Live

- Hawkins, W. (2012). *Pythagoras, the music of the spheres, and the wolf interval* [online]. Available at: <http://philclubcle.org/papers/Hawkins,W20111115.pdf> [Accessed 14 Dec. 2017].
- Holland, S., Wilkie, K., Bouwer, A., Dalgleish, M. and Mulholland, P. (2011). Whole body interaction in abstract domains. In: D. England, ed., *Whole body interaction*. Human-Computer Interaction Series. London: Springer Verlag, pp. 19–34.
- Humphreys, S., Davies, R. and Dorling, M. (2014). *CAS master teacher programme* [online]. Available at: <http://community.computingschool.org.uk/files/4560/original.pdf> [Accessed 22 Dec. 2017].
- Innes, K. (1997). Using music technology at key stage 3. In: *British educational research association annual conference*. University of York. Available at: www.leeds.ac.uk/educol/documents/000000329.htm [Accessed 14 Dec. 2017].
- John, M. (2013). *Teachers numbers fall but DfE confident on “Computing”* [online]. Available at: www.agent4change.net/policy/curriculum/2029-teacher-numbers-fall-but-dfe-confident-on-computing.html [Accessed 10 Dec. 2017].
- John, M. (2014). *More than half teachers “not confident in Computing”* [online]. Available at: www.agent4change.net/policy/curriculum/2264-more-than-half-teachers-not-confident-in-computing.html [Accessed 10 Dec. 2017].
- Johnson, T. (2006). Self-similar structures in my music: An inventory. In: *MaMuX seminar*. Paris: IRCAM-Center Pompidou. Available at: http://repmus.ircam.fr/_media/mamux/saisons/saison06-2006-2007/johnson-2006-10-14.pdf [Accessed 10 Jan. 2018].
- Kemp, P. (2014). *Computing in the national curriculum: A guide for secondary teachers* [online]. Available at: www.computingschool.org.uk/data/uploads/cas_secondary.pdf [Accessed 14 Nov. 2017].

MAMIC Goes Live

[Kochhar](#)-Lindgren, K., Schneidermann, D. and Denlinger, T. (2009). *The exquisite corpse: Chance and collaboration in surrealism's parlor game*. Lincoln and London: University of Nebraska Press.

Leyshon, A. (2009). The software slump? Digital music, the democratisation of technology, and the decline of the recording studio sector within the musical economy. *Environment and Planning A*, 41(6), pp. 1309–1331.

[Loy](#), G. (2006). *Musicmathics: The mathematical foundations of music* (Vol. 1). Cambridge, MA: The MIT Press.

Nielson, J. (1995). *10 usability heuristics for user interface design* [online]. Available at: www.nngroup.com/articles/ten-usability-heuristics/ [Accessed 2 Jan. 2018].

Paul Hamlyn Foundation. (2014). *Inspiring music for all: Next steps in innovation, improvement and integration* [online]. Available at: www.phf.org.uk/reader/inspiring-music/key-issues-challenges/ [Accessed 2 Jan. 2018].

Payne, C. (2018). *MAMIC* [online]. Available at: <https://github.com/chrispayne1/MAMIC> [Accessed 15 Jan. 2018].

Puckette, M. (2017). *Software* [online]. Available at: <http://msp.ucsd.edu/software.html> [Accessed 2 Jan. 2018].

Rogers, Y. (2011). Interaction design gone wild: Striving for wild theory. *Interactions*, 18(4), pp. 58–62.

Sandred, O., Laurson, M. and Kuuskankare, M. (2009). Revisiting the illiac suite—a rule-based approach to stochastic processes. *Sonic Ideas/Ideas Sonicas*, 2, pp. 42–46.

Sawyer, B., Forsyth, J., O'Connor, T., Bortz, B., Finn, T., Baum, L., Bukvic, I., Knapp, B. and Webster, D. (2013). Form, function and performances in a musical instrument MAKERs camp. In: *44th ACM technical Symposium on Computer Science Education (SIGCSE'13)*. Denver, CO: ACM Press. Available at:

<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.717.2615&rep=rep1&type=pdf>
[Accessed 20 Dec. 2017].

Selby, C. and Woollard, J. (2014). Refining an understanding of computational thinking. *Technology, Pedagogy and Education*, 2013, pp. 1–23.

Tenney, J. (2008). On “Crystal Growth” in harmonic space (1993–1998). *Contemporary Music Review*, 27(1), pp. 47–56.

Thorley, M. (2005). Music technology education: Who is the customer, the student or the industry. In: *Leeds International Music Technology Conference (LIMTEC)*. Leeds: University of Leeds.

Tymoczko, D. (2011). *A geometry of music: Harmony and counterpoint in the extended common practice*. New York: Oxford University Press.

Ubuntu Studio. (2017). *Ubuntu studio* [online]. Available at: <https://ubuntustudio.org/> [Accessed 12 Jan. 2018].

Wing, J. M. (2006). *Computational thinking* [online]. Available at: www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf [Accessed 14 Nov. 2017].

Wright, D. (2009). *Mathematics and music* [online]. Available at: www.math.wustl.edu/~wright/Math109/00Book.pdf [Accessed 19 Dec. 2017].

Year of Code. (2015). *What is year of code?* [online]. Available at: www.yearofcode.org/ [Accessed 17 Dec. 2017].

Notes on Contributors

Dr. Mat Dalglish has created digital musical instruments, music-related interfaces and sound installations for more than a decade. He is particularly interested in the live electronics of David Tudor,

MAMIC Goes Live

tangible user interfaces and sound synthesis. Mat is currently Course Leader for MSc Audio Technology at the University of Wolverhampton (UK).

Chris Payne is currently a doctoral student at the University of Wolverhampton and Course Leader for FdA Music Performance Technologies at South Staffordshire College. He has extensive experience as an international DJ and dance music producer for multiple record labels under the alias Casper.