Processes in Composition and Performance Leading to Performance

Catherine Hope
Edith Cowan University

THE COMPOSER AND THE MACHINE: ORGANIC PROCESSES AND MUSICALITY IN COMPUTER PROGRAMMING FOR MUSIC.

Dr Cat Hope
CREATEC
Western Australian Academy of Performing Arts
Edith Cowan University.

ABSTRACT

This paper contemplates different processes and results developed by 'programming composers' as compared to composers who use programmers to facilitate or realise compositional components of their works. Different models for the relationship between music composition and computer programming are examined, as are the outcomes for composers and performers.

For music programmers the compositional process varies according to the composer and the work they wish to create. Complex musical configurations involving sound synthesis, processing, aleatoric and improvisational approaches may be guided by conceptual ideas that do not always originate with programming skills, and can be outsourced within differing levels of collaboration. Gerald Strang's seminal 1970 essay 'Ethics and Esthetics of Computer Composition' asks if it is possible for a 'programming composer to apply similar kinds of aesthetic and analytical judgments as a composer who does not program [Strang 39]. This paper contends that things have changed, and if the act of music programming were thought of as 'musical' by all composers, it could be employed to further the timbral and structural palettes of music composition for all music. Using works of her own and her peers, and a discussion with a 'programming composer,' the author discusses some different ways to recognise musicality in computer programming.

1. INTRODUCTION

Since the introduction of electronic music, artists have been fascinated with the timbral possibilities the new 'instruments' electronics offered up. Composers have enjoyed the organisational and decision making possibilities of computers interactivity, sound synthesis, algorithmic and generative composition systems. Despite the number of ways computers can be used to make music, computers in music composition are usually employed as part of one of two processes: algorithmic composition or timbral design [Di Scipio 1]. This binary oversimplifies the role of computers in composition, as they may be employed in different levels of these processes. Sometimes the algorithmic and timbral decisions in a composition may be made by someone who does not in fact program these parameters, but rather 'outsources' any programming that may be required. This develops a kind of "within" and "without" approach to the processes of musical organization where computers are used.

I am one of those 'without' composers, I don't program, and have never had a desire to. But I use computer processing in my compositions almost all the time. This led me to wonder about any perceptible difference that may be apparent in my works, attributed to the fact that I outsource all the generative sound and score programming. I began a discussion with my collaborator in new music ensemble Decibel, Lindsay Vickery about his approaches and thoughts on having been a 'programming composer' for many years. It seems very likely, forty years after Strang's essay, that music programmer's know their tools well enough to make intuitive and artistic judgments during the composition processes they engage with. Rather than deciding which decisions 'ought' to be delegated to a computer according to some human esthetic value [Strang 41], the process has become much more intuitive and open. It begs the question, then, is there a musicality to programming?

2. COMPOSITIONAL APPROACHES

James Harley posits two main facets to compositional approaches. One is psychological, studying creativity as it is applied to music, and the other is cognitive, developing models of the mental procedures and structures used in music-making [Harley 221]. Music has often been referred to as the art closest to science [Ball 23 and others], and the involvement of computers may have well amplified that relationship, especially in regards to formalism. Musical processes can be produced using formal tools, such as algorithms or other formulae, as generative and transformative devices, yet other compositional instances call for strategies relying on interaction in order to control and qualify results and choices [Vaggione 58]. Current computer functionality deal with both of these instances easily and instantaneously, and there are plenty of programs that enable straightforward manipulations of these parameters, using graphical interfaces. This permits computer programmers to drive sophisticted musical activity that gets close to fusing Harley's psychological and cognitive mindsets. Of course, the formal rigor of a generative function provided by a computer does not guarantee by itself the musical coherence of a result [Vaggione 54], but it can be argued that as computer programming has become more commonplace, complex and refined in its application to music. It is more likely to be imbued with a musical, rather than scientific or calculated, sensibility manifest in the degree of intuition, abstraction, improvisation and responsiveness seen in music programming.

Interactive programming environments such as Quartz Composer, MaxMSP and Pure Data have proven very attractive to composers and laptop performers, as they offer an environment to compose, control and parametise music using a variety of different processes [Puckette 31]. Yet to date, these programs are usually used by composers in their own works, and rarely asked for as 'instruments' by other composers. Some 'composer programmers' use the functionality these programs bring to their compositions without being able to operate them, and outsource the programming to others. Examples include my own work, but also compositions by Pierre Boulez and Anthony Pateras, as well as many others.

2.1. Ideas out of the air

A composer may come up with an idea, and look for the best way to realise it. The skills or craft to perform the realisation of every element in the idea need not lie with the idea maker. 'Non programming composers' have handed musical interpretation of their music to instrumentalists for hundreds of years, yet have been slower to delegate to computing, unless it refers to audio engineering processes, such as mixing and mastering. The prevalence of electronic music and the adoption of interactive programming in popular music such as that by Aphex Twin, Amon Tobin, Radiohead and others has meant that the contributions of these programs are more visible. So outsourcing programming to achieve certain aims is a reasonable solution for many composers. Even better if they can outsource to 'composer programmers', who are more likely to be able to negotiate the concepts, terms and potential involved in compositional processes. By bringing a conceptual idea to a programmer, they are offering a possibility or ambition that the programmer may have never had considered for that program, thus extending their own ability on and perceptions of the software tool. In a discussion with Lindsay Vickery, he noted that

> *it's pretty easy to keep using the same tools and therefore generating the same kind of piece. It makes sense of course to keep using things that work. A fresh concept and pair of ears can definitely drive things in interesting directions though* [Vickery & Hope par 16].

SO not only is this approach beneficial to the 'non programming composer', it may offer insights to the 'composer programmer' as well.

2.2. Finding the idea in the program

Some composers 'find' an idea within a computational process. The idea and way to realise it could come from knowledge of a software capability, but could also be a conceptual construct that they bring to the program. As Vickery notes,

> *The process involves a sort of to and fro between drawings, spreadsheets (to work out the maths), the score and the software. There is often a period of tightening up where there are small changes made to all of the elements of the piece.* [Vickery & Hope par 2]

There is an important difference in these approaches. The computer becomes a tool that grows with the composer, and like the performer of an acoustic instrument – a sound or approach might be discovered, developed, cultivated and refined through practice, performances and compositions. As John Bischoff describes the creation of his work *Audio Wave* (1979-1980):

> *As I worked, I tried to peer into the behavior of the machine to see from where the next musical angle would come. My strategy was to accept the medium 'as is', using its confines as a possible avenue of discovery, rather than allowing myself to be distracted by the wish for a more perfectly plastic material* [Bischoff 79].

Working with the software in-depth offers up possibilities for its use that might not occur to another. This can create new ideas and possibilities, but can also create a cyclic expectation that is limited by the possibilities of the program itself. As Vickery notes;

> *Perhaps people who program themselves are more aware of the limitations. This can be a good and a bad thing of course – perhaps that inhibits programmers from exploring areas that are difficult or unstable or unreliable. Sometimes the solutions for*

interesting new problems take a lot of thinking about, I know I've toyed with some ideas for weeks before the eureka moment arrives. I mean in an indirect, niggling kind of thinking that goes on intermittently night and day... I think that sort of obsession is pretty hard to inspire in someone else [Vickery & Hope par 18].

3. CONCEPTUAL STARTING POINTS

Lindsay Vickery's *Antibody* (2009) employs the biological principles of mutation as a structural framework for the work. Five musical cells, stated at the beginning of the piece, are subjected to increasing 'mutation', using processes such as deletion, duplication, inversion, insertion and translocation. This mutation takes place in two layers; within the generation of the score and through a process of live audio effect, both using MaxMSP, a program Vickery has employed in his compositions for many years. The score 'parts' are written on Finale and then adjusted in real time on a laptop for the performer to read. Each of the five cells is transformed into different arrangements using the processes outlined above. In addition, the performances of the musicians reading these real time arrangements are transformed electronically

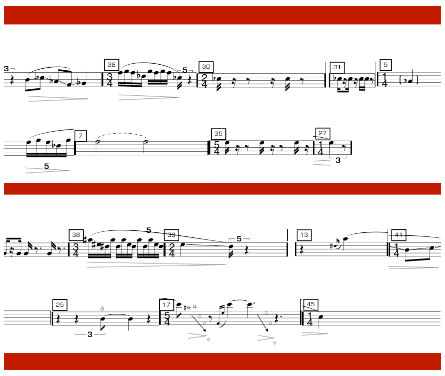

Figure 1: and excerpt showing two of the performer interfaces in Vickery's *Antibody* (2009), showing segments of music as they have 'appeared'.

using the same transformative principles for audio processing [Vickery]. These processes are ones that the program chosen to perform them, MaxMSP, does very well. Vickery's experience means that he is able to control the programs potenitial and possibilities, using them to fulfill his compositional concepts. MaxMSP enables real time processing of instruments playing in real-time and is also very good at generating aleatoric choices within parameters set by the composer. '*Antibody*' is an example of a composition where the understanding of the processes of MaxMSP offers is integrated in the very structure, scoring and sonic outcomes of the work. The plurality of layers in which the programming is involved in the work makes it very much part of its musical fabric.

In the authors work, *The Possible Stories of Harry Power* (2010), the computer has no audio synthesis or output, but is used as a kind of score generator, operating according to parameters set by the composer. Here the role of the computer programming is mostly 'back end' and the musicality of the programmer is not as vital. Yet a 'composer programmer' doing this work would be more like to understand the concepts described to them during the process of 'outsourcing'. In this work, three performers play a score written by the composer, and the computer 'listens' to their performance, using that data to write the next part of the score. The computer listens again to the performance of this new score, and creates another. The final part is provided by the composer, influenced by the kinds of scores created by the computer. These have been informed by the initial testing done during the construction of the MaxMSP/Jitter patch designed to run the score generator. Here is a concept developed by the composer, based on ideas about story telling and the importance given on the written histories (written notation) over oral ones (improvisations). Like *Antibody,* it takes its idea from an external notion and realizes with a piece of music, facilitated by MaxMSP/Jitter. Yet this work was conceived differently outside of or 'without' the program. The program was employed when it was decided the best mechanism to realise this generative procedure, some time after the concept was developed. It was not an idea that came from years of working with MaxMSP.

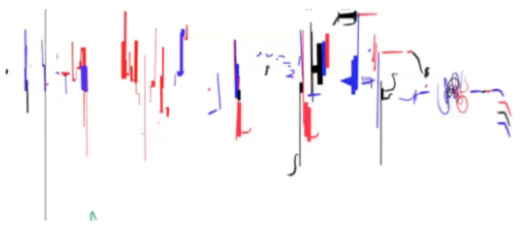

Figure 2: an excerpt of Hope's *The Possible Stories of Harry Power* (2010) showing the computer writing on the left, and the composers 'modified' writing on the right.

A 'composer programmer' is likely to try things, listening as they go – working inside the programming, creating actions, then coming out of that programming 'space' and working as the listener, perceiving their actions. This is a process common to all composers, the making/writing and playing/listening/workshopping. Improvisation offers a different system: a simultaneous making and listening on the fly. Computer programmers have become very good at this too, further bolstering a claim for 'musicianship'. Whilst a composer performs a critical act with relationships and their representations [Vaggione 60], 'programming composers' do this on many more levels, as a program is another kind of representational system that can be used within and alongside more traditional music representations. As Vickery points out;

A composition is made up of many decisions. I felt somehow constrained by having to fix certain variables in a linear score; so many possibilities have to be ignored. What I have tried to do is to leave some pathways open, allowing each performance to explore a different trajectory. There is scored materials and the ways that performers interpret them and then audio processing of their performance and how that is distributed back into the space– my aim was to allow some of these components to unfold independently of one another [Vickery & Hope par 10].

4. COMPOSING FOR, NOT FROM, COMPUTERS

Another way the 'composer programmer' may contribute to a music composition is when the computer is given a role on the score as an instrument within an ensemble. Rather than employing a programmer to help write the mechanics for a piece, the composer offers up something for the computer programmer/performer to read and interpret in a live situation. Graphic notation for electronics has become commonplace, but more often as an illustration of a prerecorded sound,

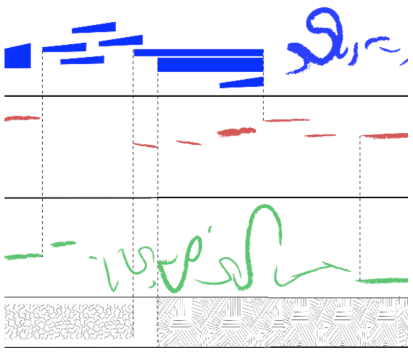

Figure 3: An excerpt from Hope's *Kuklinski's Dream* (2010) the computer part being the bottom most part.

followed for cues, volume and other control factors, rather than as a part to be 'interpreted'. This has made an important part of my own composition style, and in the illustrations below, two examples are provided. Figure 3 shows an excerpt the score for *Kuklinksi's Dream* (2010) where the computer part is notated along the bottom. The key explains three states for the computer: recording (sampling, to the left of this excerpt), playing back (to the right of the excerpt), and effected playback (not shown here). No instructions on how to effect the playback are provided; therein lay the artistry of the performer, deliberately permitted by composer.

In Figure 4, there is less freedom of interpretation involved, as the score does provide an instruction for a computer operator to transpose and extend a given moment in the strings, and route the sound into a bass amplifier. The skills here are more practical and operational, yet the quality of sound (in particular in the 'hashed' flags, denoting a distorted tone as opposed to black flags, which denote a clear tone) is very much the domain of the programmer, a timbral element the composer has given to the computer programmer to make their own. A range for the pitch is given, leaving the programmer to decide if the computer will make the choice of pitch randomly, or the program operator will decide in the live situation, or even decide beforehand.

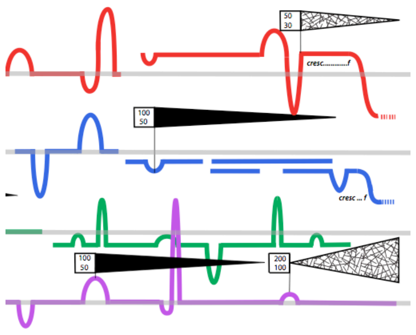

Figure 4: An excerpt of the score for Hope's *Cruel and Usual* (2011) showing the computer parts as flags (the shape describing the dynamic) with frequency ranges.

Both these samples provide an open invitation for musicality from the programmer; with their own part to perform in the score, as part of the ensemble. In a way, their artistry is amplified into the foreground of musicianship in the group, rather than the 'machine room' of the composition. Rather than programming used to set conditions for musical action, as is the case in Vickery's *Antibody,* here the musical action sets the conditions for programming. The ideas come from 'without' – outside of the programmers experience.

5. CONCLUSION

With the high level of musicality in the different situations where computer programming is featured in music, computers have been thought of as instruments in their own right for some years now. This has been propagated more by live performance (by such skilled practitioners as Robin

Fox, Kim Cascone and others) than by back end programming. However, it is less commonplace to find computer programmers as part of mixed ensembles, or programmers working to another composers brief, than you would expect. Computer music still seems separated from much contemporary classical music, sitting within its own ‘electronic music’ niche, and whilst there are examples of electroacoustic work (where electronics and acoustic instruments are combined in a group) beyond the ones discussed above, this area is a fertile one for development.

The computer in music is “a mechanism with which we interact, not a mathematical abstraction which can be fully characterized in terms of its results” [Winogard 391]. If ‘non programming composers’ can see the richness of musical possibilities that sophisticated programs offer, and treat music programmers as the musicians they really are, a huge range of possibilities becomes open to them, and computer assisted composition will move out of ‘electronic music’ and into the full realm of musical possibility.

6. REFERENCES

Ball, P. 2010. The Music Instinct: How Music Works and Why We Can’t Do Without It. London: Bodley Head.

Bischoff, J. 1991. “Software as Sculpture: Creating Music from the Ground up.” *Leonardo Music Journal,* Vol. 1, No. 1, pp. 37-40.

Di Scipio, A. 1994 “Formal Processes of Timbre Composition - the Dualistic Paradigm of Computer Music - A study in Composition Theory (II).” *Proceedings of the ICMC International Computer Music Conference*, San Francisco, International Computer Music Association. pp.1-7.

Gerzso, A. 1992. “Paradigms and Computer Music” *Leonardo Music Journal,* Vol. 2, No. 1 pp. 73-79.

Harley, J. 1995. “Generative Processes in Algorithmic Composition: Chaos and Music.” Leonardo Music Journal, Vol. 28, No. 3. pp. 221-224.

Puckette, M. 2002. “Max at Seventeen” Computer Music Journal, Vol. 26, No.4, pp. 31- 36.

Strang, E. 1970. *Ethics and esthetics of computer composition,* in Lincoln, Harry B. ”The Computer and Music.” London: Cornell University Press, p38-42.

Vaggione, H. 2001. ”Some Ontological Remarks about Music Composition Processes.” *Computer Music Journal*, Volume 25, Number 1, pp. 54-61.

Vickery, L. & Hope, C. 2011. “Discussing Programming”<http://decibel.waapamusic.com/?p=1050> [accessed 23 June, 2011].

Vickery, L. 2009 ‘Antibody’ [video] retrieved 11 May, 2011, from http://www.youtube.com/watch?v=BpVCqpMA6Jc

Winograd, T. 1979. “Beyond Programming Languages.” *Communications of the ACM.* Vol. 22, No. 7, pp. 391–401.