COMPUTER INSTRUMENT DEVELOPMENT AND THE COMPOSITION PROCESS

Victor Lazzarini

Music Technology Laboratory
NUI, Maynooth, Ireland
Victor.Lazzarini@may.ie

ABSTRACT

This text looks at the computer instrument development work and its influence on the composition process. As a preamble to the main discussion, the different types of software for sound generation and transformation are reviewed. The concept of meta-themes is introduced and explored in the context of contemporary music. Two examples of the author's computer music work are used to discuss the complex relationship between software development and composition. The first piece provides an example of such relationships in the context of 'tape' music. The second explores the use of computer instruments in live electroacoustic music. The activities of composition and instrument creation will be shown to be at times indistinguishable and mutually dependent.

1. INTRODUCTION

Computer music instruments can be generally defined as programs that allow computers to generate or transform sounds, performing as an extended musical instrument. In the past fifteen years or so, composers have been greatly interested in the use of computers in live electroacoustic music. This has been mostly due to two factors: (i) the advances in microcomputer technology; (ii) the development of graphic music programming languages, from the early Max/FTS system [1] on the IRCAM workstation, to the latest Macintosh and Windows-based software synthesis environments. The former is related to the availability of increasingly better and cheaper hardware platforms. The latter factor made programming more accessible to composers less keen on tackling text-based languages to create customised computer instruments.

In general, much of the enthusiasm generated by these developments can be attributed to the fact that computers can be treated as a multi-purpose maker of instruments, rather than a device with a limited number of applications. Trevor Wishart, a composer who also defines himself as an instrument-maker, states:

> *"Information Technology allows us to build sound-processing tools of immense generality and flexibility (...). The "instrument" is no longer definable (if subtle) closed universe, but a groundswell of possibilities..."* [2]

In response to that, the focus of the work for some composers has shifted from the traditional score-writing composition activities to incorporate the task of instrument-making into the process. In a way, electroacoustic music composers have been performing, to variable extents, the job of instrument development ever since the early days of the *Musique Concrète* [3] [4]. Nevertheless, it was only with the advent of Computer Music and Max Mathews' MUSIC series of programs [5] that the task of instrument-making was made indistinguishable from the act of composition itself. It is impossible for instance to say where instrument development stops and where 'composition' starts in Risset's *Mutations* of 1969 [6].

This article will explore how these two activities are entwined. The concept of meta-thematic processes as an important link between them is discussed. Two of the author's computer music works, *Mouvements* and *The Trane Thing*, are used as practical examples. As a background to this discussion, different types of computer software systems are examined in terms of what they offer for composers interested in instrument-making.

2. SOFTWARE FOR SOUND GENERATION AND TRANSFORMATION

The range of available computer music software for sound generation and transformation is vast. We can separate them into three levels, in terms of flexibility, programmability and generality. At the first level, we have the 'hard-wired' and semi-'hard-wired' programs: graphic software synthesizers, audio recording software plug-ins and similar systems. These tend to offer a pre-set range of functions, some provide limited programmability in terms of 'patches', mimicking outboard equipment such as old modular synthesizers. Nevertheless, the majority of these tend to limit the composer to a number of choices dictated by the musical concepts of the software designer. Their use as computer instruments is limited. Many of these programs are responsible for the dissemination of clichés and predictable uses that plague much of the music made with computers today.

At the middle level, we find the music programming languages, such as MaxMSP, SuperCollider [7], csound [8], Nyquist [9], etc. These are quite distinct from the programs discussed above by the fact that they are reasonably open-ended and programmable. They do not offer a fixed set of applications, but can be used to create computer instruments, by providing building blocks commonly known as unit generators (ugens). The level of programmability can vary quite a lot between these systems. At the higher level, we find the graphic programming packages such as MaxMSP, Pure Data and jMax. These are loosely based on the object-oriented programming model, offering classes from which objects can be instantiated and some extensibility by the use of class composition. At the middle level, we have the text-based language csound, which offers some more programmability and an extensive collection of ugens. Csound suffers slightly from the fact that it uses some outdated programming concepts (e.g. 'scores' and 'orchestras'), inherited from its predecessors MUSIC

11 and MUSIC 360. The first composition example, discussed below, uses csound for instrument development.

The best and most flexible music languages in terms of their programmability are the ones derived or extended from existing languages, such as Nyquist (LISP-based); SuperCollider (derived from SmallTalk); and Cmix/RTCmix [10] (C/C++-based). These offer a very advanced programming syntax, which can be used not only for sound generation and transformation, but also in algorithmic music composition. Generally, the level of complexity of the language increases with the programmability, so it is expected that many composers might not be prepared to tackle the lower end of this group of software. Indeed, as already noted, MaxMSP and its variants have lately become the most used of these systems, because of their user-friendliness.

At the third level, offering the best in flexibility and generality stands the 'implementation language' group of software. These comprise the C/C++ language compilers and development libraries. They offer the possibility of building instrument 'parts', ie. ugens, in form of system components (such as MaxMSP classes and csound opcodes) or complete instruments, either from scratch (using system libraries) or using audio processing libraries (such as the Sound Object Library [11]). Here the possibilities are probably extended to whatever can be defined in terms of signal processing. Composers would often be dissuaded from working at this level due to its complexity, but for those who take the challenge, a very interesting world of new possibilities is opened. The second music example discussed in this text will explore the relationship between software instrument development in C++ and composition.

3. META-THEMES AND CONTEMPORARY MUSIC

Contemporary music, since the early part of the 20th century has been characterised by the use of meta-thematic processes. In fact, this is one of the defining elements of modern and post-modern music, which sets them apart from the so-called period of common practice (Baroque to Romantic). While the traditional approach had previously been focused on thematic composition of different forms, since Schoenberg, meta-thematic techniques have dominated much of contemporary music.

Meta-themes are generative principles rather than explicit musical statements. They can be instantiated in thematic or non-thematic forms. Elaboration of meta-themes often takes place at a pre-compositional stage; however, in many cases they can arise as part of the music writing process. Examples of meta-thematic procedures are found in different styles and genres, from dodecaphonic music to Messiaen's parametrisation of musical material to aleatoric and process composition.

A composition can be based on a single meta-theme, or on a series of meta-thematic elements. For instance, in the case of serial music, there is often a single permutational principle as a generative meta-theme. A piece can also have elements with a clear meta-thematic origin combined with others that are do not have such derivation. In certain examples of process music, the basic harmonic-melodic material does not have any particular provenance (apart from being generally tonal or modal), while the rhythmic manipulation and formal development is derived from a generative principle.

Particularly important is the role that meta-thematic processes play in Computer Music. An early example of this is found in the already mentioned *Mutations* by Risset. In that piece, the guiding principle, according to the composer is that

> *"Mutations refers to the gradual transformation which occurs throughout the piece, and to the passage from a discontinuous pitch scale, at the beginning, to the pitch continuum at the last part."[12]*

This meta-thematic idea is further refined prior to the realisation of the sonic structures that compose the piece. Moreover, it serves as the support for the development of the computer instruments used in the piece. It is at the meta-thematic level that we find the basic design for both composition and instruments. This is indeed, typical of Risset and other Computer Music composers, in that the roles of composer and instrument designer become entangled. Meta-themes can operate as a link between the two activities.

A recent work by Rajmil Fischman [13] illustrates the point even more completely. In his article, a complete description of meta-thematic procedures leading to their sonic realisation through computer instruments is shown. The musical objective was to deal with the application of Quantum physics concepts in the generation of stochastic music structure with granular synthesis. A single equation (Schroedinger's Equation) was used as the basic generative principle for instrumental (synthesis & processing), sonic and musical development. This work is also significant that it points to generalised connections between a particular meta-thematic idea and their concrete instrumental (sound-generative) and musical (structure-generative) instances.

From the above discussion, it might be inferred that the use of meta-themes indicate a bias towards a total-organisation approach to composition. This is, however, not true, as it is equally possible to detect such elements in music not characterised by structuralist methods. In both Fischman's and Risset's examples, in fact, structure is dictated by sonic preoccupations rather than purely organisational ones. In addition, as hinted before, meta-thematic coherence can arise from the compositional process, even when it is not explicitly sought. Many examples of this are found in the music of the acousmatic school.

4. COMPOSITION EXAMPLE 1: *MOUVEMENTS* (FOR 8-CHANNEL TAPE)

The piece *Mouvements* is an example of the use of csound in sound and structure composition. That system was used to generate all the synthesised sounds, which were subsequently put together using a multitrack program. The fact that csound is a quite flexible tool for creating sounds was crucial in the composition process. This Section will outline the main elements involved in the development of the piece.

This piece is structured around a single principle, its *meta-theme*, which governs the synthesis procedures used, resulting in music elements that are linked together by a certain audible trait. This generative principle (and its different implementations) was developed step-wisely. The basic form for it was elaborated by modifying a Shepard-tone instrument design [13]. This instrument creates the acoustic illusion of a glissando that continuously descends or ascends without ever reaching an end point. This particular design created these glissandos by using ten sinewave oscillators tuned in octaves, constantly sliding down ten octaves from the top-most frequency. The illusion is created by using an

overall spectral envelope, in the shape of a Gaussian window, which attenuates the lower and higher ends of the glissando. The first modification made to the instrument was the elimination of the frequency glide, which created a pedal tone constantly changing in octave. In addition, the interval between each oscillator was made variable. With this modification, they could be separated by any frequency ratio, not only octaves. This instrument design became the basis on which all other instruments were created for the piece. Because they share the same principle, their sonic output will also share the same gestural and textural traits.

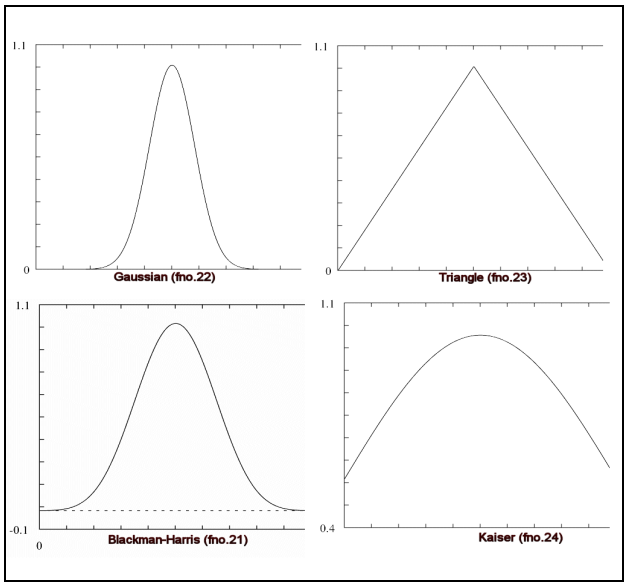

Figure 1: *The four window shapes used*

This basic design was used as the starting point for the composition of *Mouvements*. Some further modifications were made to it. The single-shape spectral envelope was substituted with a choice of four window types: Gaussian (the original), Bartlett (triangle), Blackman-Harris and Kaiser (Fig. 1). It was also made possible to vary the base frequency and the period of the envelope cycle in time, so that a moving gesture could be articulated. The sound output of each oscillator was also panned between pairs of speakers in parallel with the amplitude envelope. In addition, the direction of the envelope cycle (upwards/downwards) and a sound fade in/out time could also be set. This became the first implemented instrument used in the piece. The overall design, which will be shared between instruments, can be described as having the following characteristics (Fig. 2):

- 10 parallel audio signal generators with evenly spaced, but variable, frequency intervals;
- An overall spectral envelope (determined by the window shape) that cyclically sweeps from one end of the spectrum to the other at different speeds, emphasizing/de-emphasizing the sound of each signal generator;
- Individual spatial placement of the spectral components (also controlled by the envelope cycle).

Eleven different implementations of this design model were developed. They have the following characteristics:

1) The original one with 10 sinewave oscillators for sound generation.
2) Using a 10-piece filterbank with white noise as input; the evenly spaced frequencies were re-interpreted as the filter centre frequencies.
3) Using 10 FOF generators: This uses the base frequency (1st) as the fundamental, above which 10 formant frequencies are placed, following the original spacing rule (thus making the highest 11 times the interval over the fundamental).
4) Same as (1) except that the interval ratio is made to vary randomly 10 times/sec.
5) Same as (2) except that the amp envelope table lookup position for each signal generator is controlled by a random number generator. In (2) [and all other instruments], this was controlled by a constantly cycling variable going from start to end or end to start of the envelope.
6) Using a 10-carrier FM set-up. The evenly spaced frequencies are interpreted in two ways: (i) for carriers 2-5, they are effectively formant frequencies; (ii) for carriers 6-10 they are taken as the carrier frequency, directly.
7) Same as (6), except that only carriers 2-3 are using formant frequencies.
8) Same as (6), except that all carrier frequencies are set directly to the evenly spaced frequency values.
9) Same as (3) with same randomised and variable parameter (FOF grain size and 'local' envelope).
10) Same as (9) with a different method of randomisation.
11) Same as (10) with some more variable parameters.

The different implementations listed above were developed in response to musical needs. For instance, as the start gesture for the piece was being composed, it was felt that a richer spectral content would be more suitable for it. This was then realised by using filtered white noise instead of sinewaves, giving origin to the first variant on the basic model. In the middle section, the continuous textures were supposed to be interrupted, so an instrument that could generate more percussive sounds was developed. By keeping to the general design and varying the sound synthesis technique, a variety of interconnected musical elements were generated. Depending on the choice of parameters (spectral envelope shape, sweep rate, etc.) and instrument design, different types of gestures and textures can be created: spectral 'arpeggios', glissandos, drones, chords, timbral mutations, etc..

Here, it is clear that instrument design is crucial to the composition process, being, in fact, seamlessly integrated to it. It not only responds to structural needs, but also informs the creative activities by providing the different ways of articulating ideas. The use of csound was also decisive, because it provided a flexible platform for the development of these instruments.

In terms of its overall structure, this piece uses conventional composition techniques such as variation/development in an extended manner. There is no such thing as a theme, but a working principle, a meta-theme which that links all sounds, textures and gestures in the piece. It is used to generate an overall sense of unity, without recourse to traditional compositional methods. Structural ideas similar to this one are found in many examples of computer music. In fact, this method of work was probably born out of the possibilities brought on by computer instrument development. *Mouvements* was composed as a tribute to J C Risset.

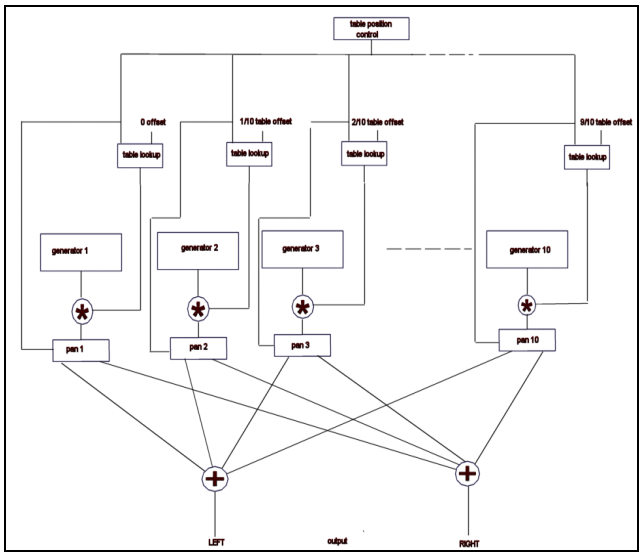

Figure 2: *The basic instrument design*

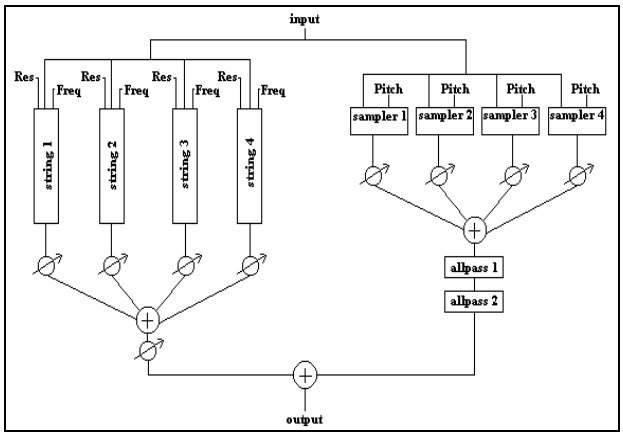

Figure 3: *Instrument flowchart*

5. COMPOSITION EXAMPLE 2: *THE TRANE THING* (FOR TENOR SAXOPHONE AND COMPUTER)

The composition of *The Trane Thing* is an example of how research activities in computer music can be linked to an artistic output. The piece was composed as a duet for saxophone and computer instrument. This was developed as a stand-alone piece of software created using the Sound Object (SndObj) Library, a set of programming tools developed as part of the author's research. The library was used to provide the audio processing aspect of the instrument. The graphical user interface was built using the Microsoft© Foundation Classes framework. The instrument was custom-built to the needs of the composition. The use of software development tools for its creation was essential, because it provided a way to shape the instrument to match exactly the needed specification.

A basic instrument was created and these ideas were tested with some saxophone material. Slowly, the sonic shape of the piece started to crystallise. It was soon realised that more 'landscape' sounds were needed to involve the saxophone. Four sampling operators were then included to capture and replay some saxophone material. These completed the original goal of immersing the saxophone in a pool of sound. They also provided the means of capturing improvised gestures, something that became a compositional interest from the very start of the work. In fact, improvisation turned out to be one of the defining elements of the score, as this grew to become a tribute to the late American saxophonist John Coltrane (Figure 4). The contemplative, slightly modal, sound of the piece was certainly influenced by his music.

The instrument design is relatively simple and uses basically two main components, a string resonator and a sampling operator, as seen in Figure 3. The main idea was to create layers of pedal sounds behind the saxophone lines that would constantly change according to the notes that composed those lines. The string resonators were ideal for this purpose, because they could be tuned to structurally significant frequencies and would react to them and to their harmonics. A decision was made to limit the instruments to four strings, for the practical reason that more strings make the result more undistinguished. The use of several resonators creates a diffuse reverberation field. This, of course, was not the intended result, as the main job of the string resonators was to create drones and sustained harmonies.

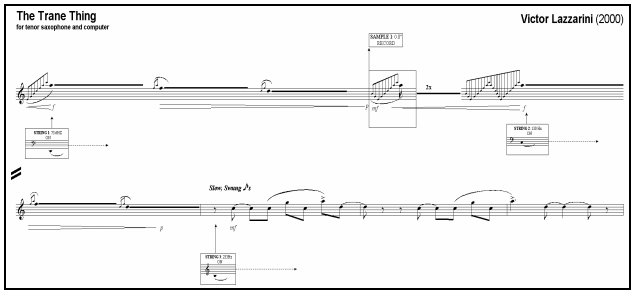

Figure 4: *Excerpt from the score of The Trane Thing*

The user interface was also designed in parallel to the piece composition, according to what actions were required by the score. A few extra elements (frequency and pitch adjustments etc.) were also included, but they did not find a specific use in the piece. The instrument window is shown on Fig. 5. An important point was to make it playable, so the controls were made very accessible and easy to use. MIDI control was not included, in order to keep the required set-up to a minimum, so that the piece could be performed more often. It only requires a computer (laptop or desktop) running Windows operating system and a reasonably good soundcard. The fact that it has been performed quite often (in Ireland and abroad) is in part due to these careful considerations.

This work shows an example of how instrument development fits into the composition process. On one hand, it influences how the music is composed by providing part of its sonic characteristics, on the other it is influenced by the requirements of the score. In fact, these two elements of the composition work, score writing and computer programming, are mutually dependent. As they evolved together, there is a special type of integration being

forged between the music and its instrumentation. This is something that would not be otherwise possible. The ability of a composer to perform both tasks, score-writing and programming, is essential. The musical result is then solely in his hands.

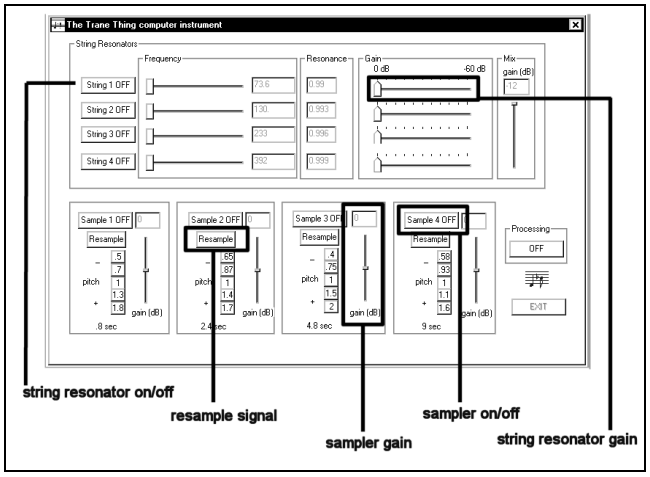

Figure 5: *The Trane Thing instrument and its main controls*

This piece was not written explicitly using a meta-thematic basis developed at a preliminary stage. However, arising from the composition process itself, there are meta-thematic elements that define the soundscape of the piece: use of Coltrane-like melodic fragments, the resonating environment, the use of open-end sections and audio capture/replay. Again, these form a groundwork that connects the activities, of computer instrument development and composition.

6. CONCLUSION

This text has discussed some aspects of the use of computer instruments in music composition. It was shown that an important element of contemporary music composition is the concept of meta-thematic development. In Computer Music, this can serve as basic link between computer instrument design and music structure.

Using two examples, the task of instrument development and its place in the composition process were explored. The first example discussed it in relation to tape music composition and the second in relation to live electroacoustic music. It was shown that the activities of composition and instrument creation could be, in some situations, indistinguishable and, in others, mutually dependent. This discussion also points to the emergence of a new kind of composer, versed not only in the traditional aspects of composition, but also in signal processing and computer programming.

There are many further issues relating to the relationship between instrument-making and composition. These range from technical to educational, aesthetic and analytic ones. Not much attention has been given, for instance, as to how computer instrument development should be taught in the ever-expanding third-level music technology area. Similarly, studies have so far only scratched the surface in relation to the aesthetic implications of a music practice that incorporates instrument-making by default. It is hoped that research in this area will be instigated by further work from music scholars, educators and composers. In such circumstances, it is expected that some of these questions will be addressed more thoroughly in future research.

7. REFERENCES

[1] Puckette, M., "Max at Seventeen," *Computer Music Journal* 26 (4), MIT Press, Cambridge, Mass., pp. 31–43, 2002.

[2] Wishart, T., *Audible Design.* Orpheus the Pantomime, York, 1995.

[3] Manning, P., *Electronic and Computer Music.* Clarendon Press, Oxford, 1993.

[4] Chadabe, J., *Electric Sound*. Prentice-Hall, Englewood Cliffs, NJ, 1997.

[5] Mathews, M., *The Technology of Computer Music*. MIT Press, Cambridge, Mass., 1969.

[6] Dodge, C, Jerse T., *Computer Music*. Schirmer, New York, 1985.

[7] McCartney, J., "Rethinking the Computer Music Language: SuperCollider," *Computer Music Journal* 26 (4), MIT Press, Cambridge, Mass., pp. 61–68, 2002.

[8] Boulanger, R (ed.), *The CSound Book.* MIT Press, Cambridge, Mass., 2000.

[9] Dannenberg, R., "Machine Tongues XIX: Nyquist, a language for composition and sound synthesis," *Computer Music Journal* 21 (3), MIT Press, Cambridge, Mass., pp. 50–60, 1997.

[10] Garton, B., Topper, D., "RTCmix: Using CMIX in Real-Time," `http://www.music.columbia.edu/cmix/rtrealtime.html`.

[11] Lazzarini, V., "The Sound Object Library," *Organised Sound* 5 (1), Cambridge Univ. Press, Cambridge, UK, pp. 35–49, 2000.

[12] Schrader, B., *Introduction to Electro-Acoustic Music*. Englewood Cliffs, NJ, 1982.

[13] Fischman, R., "Clouds, Pyramids and Diamonds: Applying Schroedinger's Equation to Granular Synthesis and Compositional Structure," *Computer Music Journal* 27 (2), MIT Press, Cambridge, Mass., pp. 47–69, 2003.

[14] Risset, J. C., *Introductory Catalogue of Computer-Synthesized Sounds*. Bell Tel. Labs., Murray Hill, NJ, 1969.