The Smart Controller

Angelo Fraietta

started playing guitar at the age of 15 while completing an electronics technician apprenticeship in the Royal Australian Air Force. Since that time, music and electronics have been an integral part of Angelo's creative being, leading to the recently developed Smart Controller during his doctoral research at the University of Western Sydney.

For further information, please contact Angelo Fraietta. URL: <http://www.users.bigpond.com/angelo_f/Guitar/InternetBio.htm>.

Instrument building today is not limited to acoustic instruments. It has opened up to include designing and building instruments for performances and sound installations that encompass electronics and software engineering (Bandt, 2001). Composers, for example, often collaborate with engineers (Arveiller, 1982) in order to create interactive and responsive environments, using sensors that detect changes in the physical environment, such as light and movement (Bandt, 2001; Mustard, 2002). For example, in 1997 I was involved in two collaborative projects: the Laser Harp by Alex Cockburn - an instrument that allows a performer to break laser beams with the body; and the Virtual Drum Kit by Guy Robinson, which enables a performer to play an invisible drum kit. The combined cost of the parts required to interface both these instruments to a computer cost less than ten dollars. This does not include the parts that actually made their instruments work. This was due to the fact that I was using controller keyboards as control voltage (hereafter CV) to MIDI converters. Although extremely bulky, the controller keyboards served the purpose of both projects. However, I would approach the task differently today, if on a similar budget.

Commercial CV to MIDI controllers are now available ranging from AUD 230 for one of my MIDI controllers, up to and exceeding USD 600 for an I-Cube (Infusion Systems, 1998). Ross Bencina, on the other hand, has provided a brilliant on-line resource for developing your own CV to MIDI converters using PIC micro-controllers (Bencina, 2002). Today it is possible, therefore, to build a very limited device for about AUD 50. But in making a significant monetary saving, one still has to learn how to program the device – an investment of time.

Building or Buying a Device?

The first question you must ask yourself is whether you should build or buy? Building a new instrument is a valid creative process in itself. However, it can be frustrating due to the enormous amount of work that must be accomplished before being able to produce an acceptable performance with the new instrument (Burt, 1999; Arveiller, 1982). In some cases, a part of the instrument only serves to perform a function that could be replaced with a single component. Designing this part might be seen as a waste of creative energy that could be served better on another part of the instrument. Deciding on what to build and what to buy is often determined by what one is interested in doing, and by the amount of time or money available for investment.

The software allows users to create musical textures and events by piecing virtual objects together with patch cables on the computer screen ...

Getting Parts at the Right Price

Obtaining the required parts at the right price can be achieved by collaborating with someone who has the parts you want. Alternatively, one can use obsolete or recycled parts, which in some cases are better than the newer parts. In the standard situation, the person you are buying parts from wants your money, and in turn, you want their parts. There are, however, ways that you can obtain useful parts at a very low price – and sometimes for free. For example, Timothy Opie used one of my experimental CV to MIDI converters in his Poseidon device instead of designing his own, because his area of interest lay in granular synthesis – not MIDI development – and he obtained the device at low cost (Opie, 2002). MIDI was simply a mechanism that he used to effect control of the granular synthesiser

within his instrument, and understandably, he had no interest in developing a CV to MIDI converter. For me, as the supplier of the CV to MIDI converter, the advantage of providing the device at a low cost was offset by the fact that I would have someone actively using and testing the device at an early stage of its development, thus providing invaluable feedback (MacCormack, 2001).

Another method for obtaining parts is through a recycling approach. In some cases, old and obsolete parts can be more useful to a particular situation than brand new parts. My first instrument, the Electronic Flying Vee, was a university project. It was constructed entirely from parts found in my garage. (The University of Western Sydney offers a course in music technology where students are required to create a music instrument from parts that they find). Before going to university, I was an electronics technician, and subsequently, I was able to create a simple electronic instrument from old parts.

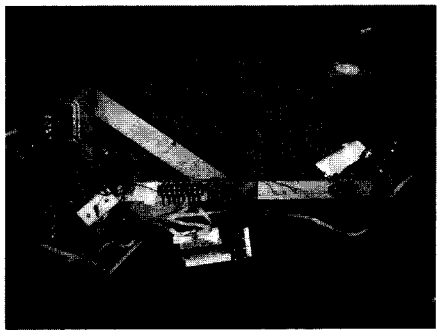
The instrument was an imitation guitar, which was played by pressing an old electric guitar string against a series of clout nails on the neck of the instrument. The pitch was determined by which clout the string was pressed against – similar to the way the pitch on the guitar can be changed by pressing the string on a higher or lower fret. The parts included a circuit board from a box of practice soldering boards, which was used to hold the electronic parts; a wooden crate that came with my new washing machine became the instrument frame. I also used various electronics components from an old amplifier that I never got around to fixing, as well as various other bits of hardware, such as light switches, nails, and wire. Although this approach might seem ad hoc, the approach is both serious and valuable, which is shown from my most recently developed Smart Controller.

The Smart Controller is an integrated system that enables composers to create musical structures by connecting virtual objects with virtual patch cables. One creates music using programmable logic control. Smart Controller was based on my Algorithmic Composer program, which is similar to Max for Macintosh. I will briefly describe the programming paradigm for this type of software.

The software allows users to create musical textures and events by piecing virtual objects together with patch cables on the computer screen, similar to the way a person might join pieces from a Meccano set to create a truck, train, crane, etc. With the Meccano set a truck might be created by joining metal panels with bolts, adding wheels, windows, motor, etc., which is just putting together pre-fabricated materials in a meaningful way to produce a desired result. Algorithmic Composer allows the user to create in a similar way by connecting counters, delays, metronomes, flip-flops, calculators and other such devices. The user can interact with the program by using the mouse and / or a MIDI input device. The computer sound card or an external MIDI sound engine produces the sound. Algorithmic Composer does not use a notation-based representation, and so a person does not have to be able to read music to use the program.

The Smart Controller is similar to Algorithmic Composer with the exception that it does not require a desktop or laptop computer in performance. The device is programmed remotely through the use of a patch editor, which is an independent computer application that simulates and communicates with the hardware. The Smart Controller responds to input CV and MIDI messages, producing output CV and MIDI messages (depending upon the patch). The Smart Controller is a stand-alone device, a powerful, reliable and compact instrument that is capable of reducing the number of electronic modules required. It does not require a laptop computer in a live performance situation. Anne Norman, for example, (see her article in this issue) is currently using a Smart Controller in her Bell Garden installation (Norman, 2003).

Designing and building the Smart Controller may at first appear to require a substantial budget for software and hardware tools, however, the cost was reduced significantly because I chose to use recycled hardware and low-cost software including educational licenses, freeware, and shareware (Fraietta, 2001).



One of the recycled parts I used was a 386 personal computer (hereafter PC) donated by a friend, David Gibbins. The 386 was used to simulate the final hardware device. I chose to run the Real Time Executive for Multiprocessor Systems (hereafter RTEMS) as the real time operating system because the source code is free, the development tools are free, and there is free support (OAR, 1988). My intended target for the final device was the i386EX embedded controller which, like the 386 PC, did not have an inbuilt math co-processor (Brey, 1998). By removing the math co-processor from the 386 PC, I would be able to determine the performance I could expect from the i386EX. During the development, I found that the floating point libraries provided with the RTEMS tools did not compile properly, and were actually making calls to the co-processor, which in turn crashed the system. I was able to determine this by reinserting the co-processor and running the code. If I had used a 486 computer instead of the older 386, I would not have encountered this problem because a 486 has an inbuilt co-processor (Triebel, 1988). The problem would not have become apparent until implementing the embedded controller, at which time the problem would most likely not have been diagnosed correctly.

Keep At It!

As stated earlier, it can be quite frustrating trying to develop a new electronic instrument, particularly when it involves writing software. I was working on the Smart Controller full time for nearly eighteen months before

continued on page 49

Angelo Fraietta

continued from page 23

generating my first MIDI message from a sub-module, and then a further six months before I got the unit working together. I commenced work on the project in June 2000 and my first performance with the instrument was not until September 2002. Part of the success of the instrument was that I abandoned the old Waterfall Lifecycle approach to design (Royce, 1970) and moved to the Unified Process (Jacobson, Booch & Rumbaugh, 1999). In the Waterfall approach, everything must be designed before you start – change in design is equivalent to failure (Larman, 2002). The Unified approach, however, embraces change as a part of the development process. Using this approach, I was able to develop sections that could exist by themselves outside the project. For example, I am able to provide low-cost MIDI controllers that were originally developed as the input / output module of the Smart Controller. Also, Quikscribe has used my streaming mechanism, developed for communicating between the Patch Editor and the hardware simulator, as the core element of their Active Document Authoring System™ (Quikscribe, 2003). What is quite humorous about the whole project is that I received a bit of flack on the Australian Computer Music Association newsgroup about my project – some stating that this sort of project was time consuming and without much benefit. I have now sold a swag of my MIDI controllers (both in Australia and abroad), and licensed software components originally created for a musical instrument. The bottom line is this: listen to your critics but don't take what they say to heart. Instead, look at their criticisms objectively. You can do one (or all) of the following: if they are correct, make your product better; if they are wrong, reject their opinion, or just ignore them – they probably don't understand or suffer from 'tall poppy syndrome' – it's your project so don't let anybody spoil your fun.

References

Arveiller, Jacques. 'Comments on university instruction in computer music composition.' *Computer Music Journal* 6, 2, 1982, pp. 72-78.

Bandt, Ros. *Sound Sculpture : intersections in sound and sculpture in Australian artworks*. St. Leonards, NSW: Craftsman House, 2001.

Bencina, Ross. 'DIY MIDI Controllers using PIC Microcontrollers and Basic Stamps'. Accessed: 28 October 2002. <http://www.audiomulch.com/MIDIpic>.

Brey, Barry B. *Embedded controllers: 80186, 80188, and 80386EX*. Upper Saddle River, NJ: Prentice Hall, 1998.

Burt, Warren. 'An Email Interview with Warren Burt'. Interview by Greg Schiemer. *Chroma*, 25, 1999, pp. 3-6.

Fraietta, Angelo. 'Smart budgeting for a Smart Controller', in *Proceedings of Waveform 2001: the Australasian Computer Music Conference*. University of Western Sydney, 2001.

Infusion Systems. *I-Cube System Manual*, 1998 Accessed: 5 March 2003. <http://www.infusionsystems.com/support/icubex-111-manual.pdf>.

Jacobson, Ivar., Booch, G. and Rumbaugh, J. *The unified software development process*. Reading, Mass: Addison-Wesley, 1999.

Larman, Craig. *Applying UML and patterns: an introduction to object-oriented analysis and design and the unified process*. 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2002.

MacCormack, Alan. 'Product-development practices that work: How Internet companies build software'. *MIT Sloan Management Review* 42, 2, 2001, pp. 75-84.

Mustard, Jonathan. 'Correlating movement in space to the parameters of sound', in *Form, Space, Time: the Australasian Computer Music Conference*. RMIT, 2002.

Norman, Anne. 'Garden Bells'. Accessed: 12 March 2003. <http://home.vicnet.net.au/~amncrow/PPBells.html>.

Online Applications Research Corporation. Real Time Executive for Multiprocessor Systems 4.5. 1988.

Opie, Timothy. 'Granular synthesis: experiments in live performance', in *Proceedings of Form, Space, Time: the Australasian Computer Music Conference*. Royal Melbourne Institute of Technology, 2002.

Quikscribe. 2003. Accessed: 17 April 2003. <http://www.quikscribe.com.au>.

Royce, Winston W. 'Managing the development of large software systems', in *Proceedings of IEEE WESCON*, 1970.

Triebel, Walter A. *The 80386, 80486, and Pentium processors: hardware, software, and interfacing*. Upper Saddle River, NJ: Prentice Hall, 1998.

■