Modernising musical works involving Yamaha DX-based synthesis: a case study

JAMIE BULLOCK and LAMBERTO COCCIOLI

Birmingham Conservatoire, University of Central England, Birmingham B3 3HG, UK
E-mail: jb@integralive.org

In this article we describe a new approach to performing musical works that use Yamaha DX7-based synthesis. We also present an implementation of this approach in a performance system for *Madonna of Winter and Spring* by Jonathan Harvey. The Integra Project, 'A European Composition and Performance Environment for Sharing Live Music Technologies' (a three year co-operation agreement part financed by the European Commission, ref. 2005-849), is introduced as framework for reducing the difficulties with modernising and preserving works that use live electronics.

1. FROM SILENCE

The Harvey Project (www.conservatoire.uce.ac.uk/harvey) began in 2002, when Birmingham Conservatoire's Thallein Ensemble performed Jonathan Harvey's *From Silence*, composed in 1988 and written for soprano, six players, and live electronics. It utilises an Akai S-900 sampler (with keyboard controller), Yamaha DX7 IIFD, Yamaha TX816 (with keyboard controller), two SPX 90 II effects units, and a Macintosh computer for control. When the work is performed in its original form, it requires three technicians to operate the electronics.

In the 2002 performance, Lamberto Coccioli and Simon Hall developed a new system for performing the piece, where all of the synthesis and sample playback was done using software running on a Macintosh computer. The two external effects processors were replaced by the internal effects from a Yamaha digital mixer. This system had the advantage that it negated the requirement for near obsolete external synthesis and sampling hardware. It also meant that all performance data required for the work became available on modern digital media, thus facilitating ease of data transfer and backup.

Multiple instances of Native Instruments' FM7 software, hosted in Max/MSP, were used to emulate the DX7 and TX816 synthesizers, and an Akai S-5000 was used for sample playback. This system reduced the set-up time for rehearsals and performances, and required only two technicians. The system also had the advantage that it used generic hardware and software that could be reused for other works, making the purchase of such a system more desirable for ensembles wishing to acquire the necessary tools for performing a range of repertoire works involving technology. Since FM7 is available for both Macintosh and Windows PCs, it was also a good choice in terms of cross platform support. However, there is currently no support for FM7 on Linux operating system.

1.1. Commercial solutions

One disadvantage of the software-based system used for *From Silence* is that it entails significant initial cost if the performing ensemble wishes to obtain the resources. Assuming that an ensemble wanted to perform the work, and already had a mixing desk and means of providing reverberation, the cost of performing the work would be approximately £1,580 (see Table 1 for a breakdown of costs). To hire the equipment originally used to perform the work would cost £360 (see Table 2), assuming that three days' hire would be required for a performance. It is clear from these prices that an ensemble would need to use the equipment for the modernised version (or similar works) more than four times to make it a viable option financially.

Whilst the modernised system provides benefits in terms of providing the tools for performing a variety of works involving Yamaha DX-based synthesis, it is

Table 1. Purchase costs for performing *Madonna of Winter and Spring* (items required by both systems such as the Macintosh computer, have been omitted). Prices correct at time of writing.

Item	Cost(£)
FM7 Software	219 (source: arbitermt.co.uk)
Max/MSP Software	289 (source: cycling74.com)
Fatar SL 990 Controller keyboard	239 (source: dv247.com)
RME Fireface	899 (source: dv247.com)
MIDI interface (common to both systems)	−70
Total	1576

Table 2. Hire costs for performing *Madonna of Winter and Spring*.

Item	Cost(£/3days)
Yamaha DX7-IIFD@35/day	105 (source: studiohire.net)
Yamaha TX816@50/day	150 (source: studiohire.net)
Akai S-900@35/day	105 (source: studiohire.net)
(M-Audio MIDISPORT interface)	(70)
Total	360

entirely reliant on commercial software. The implications of this are as follows:

(1) Commercial software is usually closed source, meaning that the source code used to write the software is not available for the user (or others) to modify.
(2) Commercial software is subject to profitability, and if companies become bankrupt, the software usually ceases to be developed.

Max/MSP provides a partial solution to these problems in that patch data are stored in plain text format. Max/MSP also has the option to export patches as runtime executables, so that they can be used without the Max/MSP environment. However, these runtime packages provide no editing facilities other than those provided by the patch, and the patch itself cannot be edited once it has been 'encapsulated' in executable form. The FM7 synthesis application supports System Exclusive format, which has an open and widely used specification, so although the application is not free, it places no restrictions on the longevity of the patch data itself.

1.2. Open source alternatives

In his paper, 'New public-domain realizations of standard pieces for instruments and live electronics', Miller Puckette (co-developer of Max/MSP) outlines a method and rationale for performing canonical works 'without resorting to special hardware or proprietary software (Puckette 2004). He describes how the live electronics elements for four historically important works were revised and modernised using PD (Pure Data) software running on a Linux PC. Puckette discusses the practical need for more cost effective and widely available performance technologies, and suggests that realisations based on open-source solutions might be 'longer lasting than previous realisations have been'.

We therefore decided that for any future attempts to migrate works from near-obsolete to modern equipment, open-source software and open standards should be used and developed. This decision was based on the notion that the initial cost of developing such solutions would be outweighed by an absence of cost to the end user. We also considered that the use of open-source solutions would provide a level of freedom over design and implementation not offered by commercial software.

2. MADONNA OF WINTER AND SPRING

Madonna of Winter and Spring by Jonathan Harvey was the second work to be adopted for modernisation. It is a work for orchestra, incorporating synthesizers, a sampler, a ring modulator and live spatialisation (see Figure 1). It lasts approximately 37 minutes and uses a Yamaha TX816 for synthesis, a DX1 for controlling the TX816 and synthesis, and an E-mu II sampler for sample playback.

In order to provide a cost-effective solution that would be robust enough to support a range of other works using similar technologies, we proposed that a number of generalised modules would be developed, and then tailored to the specific needs of the work. We decided early on that where possible, any tools used or developed for *Madonna of Winter and Spring* should be available or made available online under an open-source software licence (such as the GNU GPL). We considered that this would improve accessibility and encourage participation in further development efforts.

In order to meet these needs, the Pure Data (PD) real-time graphical programming environment was chosen as the core system for providing basic processing of audio and control data. Pure Data has the advantages that it is open source (under a BSD licence), and truly multi-platform – having been written in C and TCL/Tk with portability in mind.

2.1. DSSI

Since there are currently no built-in facilities in PD for emulating DX-based synthesisers, an open-source means of achieving the necessary DX emulation was sought. The most appropriate solution was the Hexter DSSI (DSSI Soft Synth Instrument) plug-in written by Sean Bolton as part of the DSSI project (dssi.sourceforge.net). Hexter is licensed under the GPL, supports MIDI System Exclusive (SysEx) loading of DX7 patches, and has accurate voice-level emulation of the Yamaha DX7 synthesizer. All Hexter voice parameters are programmable using SysEx messages. When the project began, Hexter only had limited performance parameter support, and no LFO functionality; however, since the software was open source, we decided that these features could be added at a later point if required.

The Hexter software consists of a synthesis plug-in written in C, and a graphical user interface written in C using the GTK+ API. It requires a DSSI host in order to run. DSSI is an API (see Figure 2), which is designed for writing software synthesis plug-ins with graphical user interfaces (similar to VSTi). It is based on LADSPA

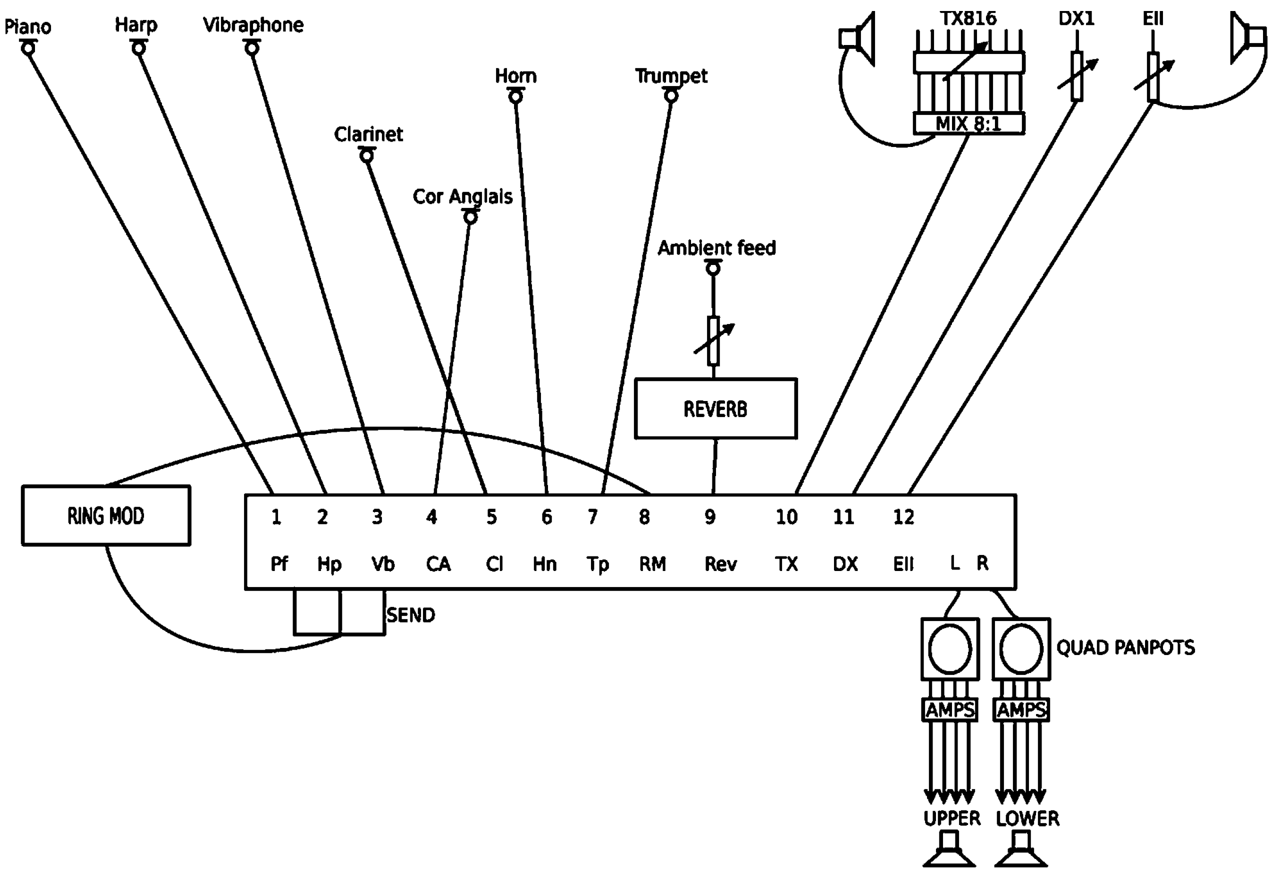

Figure 1. The original live electronics schematic for *Madonna of Winter and Spring.*

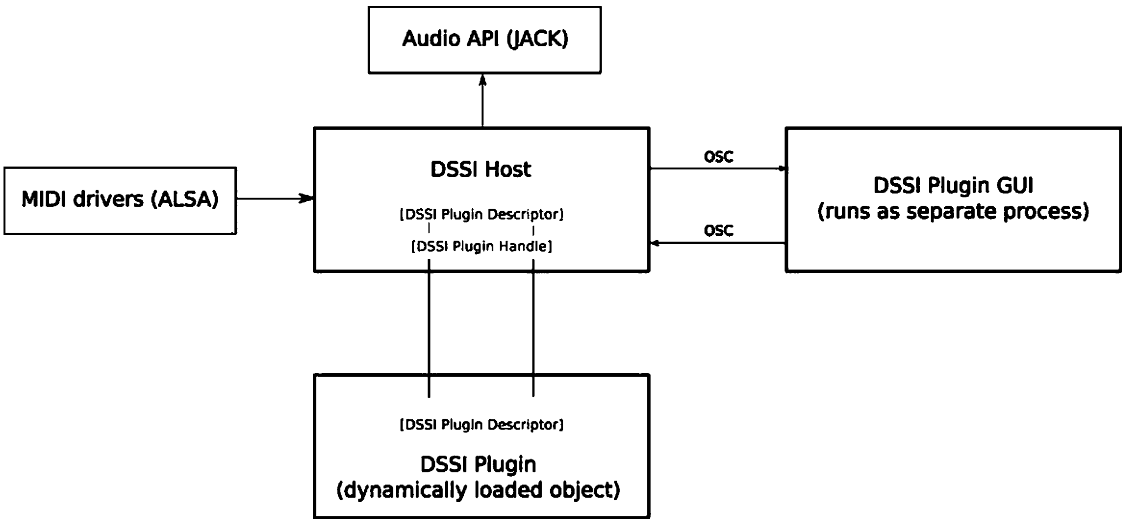

Figure 2. A conventional DSSI configuration. The Host accesses functions in the plug-in through its descriptor. Control data and audio buffers are passed to the plug-in for processing, and then output to the audio system.

(Linux Audio Developer's Simple Plug-in API), which it uses for communicating audio and LADSPA control data in memory. DSSI also adds specifications for communication between a plug-in and its GUI via OSC, and provides other facilities such as runtime configuration setting, program selection and program discovery. Typically a host will obtain a handle from the plug-in library file (a 'shared object'), and then use that handle to obtain the address of a descriptor function. Successive (indexed) calls to this descriptor function each yield a pointer to a data structure of type DSSI_Descriptor, which provides access to the API functions supported by each plug-in within the library, as well as each plug-in's LADSPA descriptor.

The DSSI header file is available under the GNU Lesser General Public Licence and a number of hosts have already been developed for Linux and Mac OS X. DSSI is a technology that already has the interest of several leading research centres; for example, Arumi and Amatriain state that their own CLAM environment (which already has LADSPA support) 'provides visual tools for rapid prototyping applications with graphical interfaces, [and] these applications are very suitable to be DSSI plug-ins' (Arumi and Amatriain 2005). Linux audio developer Dave Phillips also writes that 'DSSI has much to recommend it, including support for LADSPA and an interface for plug-in instruments (a la VSTi plug-ins)' (Phillips 2005).

There is currently a diverse range of DSSI plug-ins available including plug-ins for playing soundfonts (a sample format devised by E-mu and Creative Labs), and emulating a variety of classic analogue and digital synthesizers. We therefore decided that although there were existing hosts that PD could control using MIDI or OSC, it would be a worthwhile investment of time to write a DSSI host that operated as a PD class. Using this approach facilitates a more seamless integration between PD and DSSI/LADSPA plug-ins.

2.2. Madonna of Winter and Spring system implementation

The model for the *Madonna of Winter and Spring* system (see Figure 3), was to create TX816 and DX1 synthesizers as abstracted modules inside PD. We also considered that the EMU samples could be converted into a soundfont bank, and played back using FluidSynth-DSSI, a DSSI plug-in version of the FluidSynth soundfont software (originally by Peter Hanappe). Again this could be abstracted and presented to the user as an Emu-II module inside the PD environment.

2.3. 'dssi~'

'dssi~', a DSSI host for Pure Data is now at release version 0.9 (time of writing), and has complete DSSI API support. It was originally based on code from jack-dssi-host by Steve Harris, Sean Bolton and Chris Cannam. It now has LADSPA plug-in support, and includes code from the 'plugin~' PD external by Jarno Seppänen. This means that the full range of LADSPA plug-ins can also be used for a variety of DSP tasks within PD. These include waveform generation, filtering, reverbs and delays, and FFT-based processing. One advantage of using 'dssi~' for these tasks is that plug-ins loaded with 'dssi~' are multiply instantiable, meaning that any number of instances of a particular plug-in can be hosted with a single 'dssi~' object. This is significantly more efficient, and easier to manage than having multiple instances of 'dssi~' hosting the same plug-in.

'dssi~' has been designed to be efficient and easy to use. For example, the ability to load a plug-in simply by specifying its name as an object creation parameter has been added. This was achieved by adapting generic LADSPA loader code written by Richard W. E. Furse. Support is also available for listing all available plug-ins, showing information about a particular plug-in, and dynamically loading and unloading plug-ins at runtime. All control data (including note data) is sent to the leftmost inlet, and all output control data is sent to the leftmost outlet. The other inlets and outlets are built dynamically depending on how many instances have been invoked, and how many audio rate inlets and outlets the plug-in has for audio and/or control. Control and configuration data can be sent to one specific plug-in instance, or to all instances at once. Arbitrary configure (key and value) pairs can be sent to DSSI plug-ins, allowing patch loading and other 'configure' facilities. DSSI GUIs can be shown or hidden by sending 'dssi show' or 'dssi hide' to the relevant instance or all instances.

2.4. Hexter

Having constructed a basic DSSI host for PD, some tests were conducted to establish the suitability of Hexter for emulating the TX816 and DX1 as used in *Madonna of Winter and Spring*. We conducted a thorough analysis of the synthesizer parts (and related

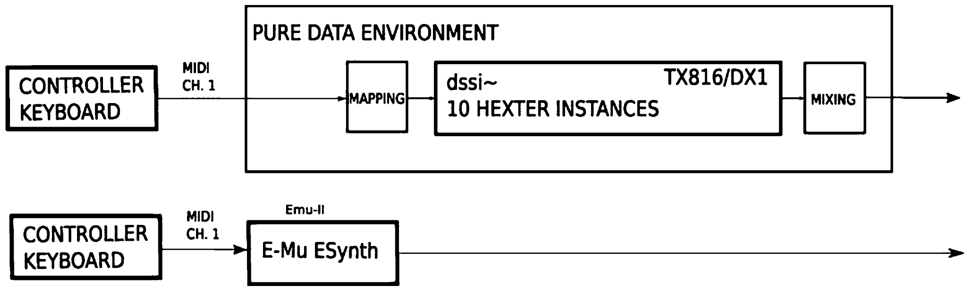

Figure 3. The current modernised version of the synthesizer elements of the live electronics for *Madonna of Winter and Spring*.

instructions) on the score, and noted any 'special' requirements. All of this information is documented on the project weblog. Analysis revealed that the piece required the following facilities: portamento (type not specified), real-time control over the 'Frequency Coarse' parameter of operators 1–6, vibrato, and multiple pitch bend ranges.

We decided that the pitch bend ranges for each module of the TX816 could be mapped using PD and that no requirement was needed to do this using Hexter (although Hexter now has facilities for this). Frequency Coarse control was available in Hexter, but only via SysEx, and would only take effect for notes played after the parameter value change. Portamento, and LFOs for amplitude and frequency modulation had not yet been implemented.

We therefore approached Sean Bolton, the author of Hexter, to establish whether he would be willing to contribute any additional functionality to the software. Since then, Sean has added many of the required features in collaboration with Birmingham Conservatoire. For example, in order to correctly emulate the LFO behaviour of the original DX7, we took measurements for LFO parameters by using a variety of PD patches to play test sounds on a Yamaha TX7 (DX7 module), and analyse the output. These results (available on the project weblog), were provided to Sean Bolton, who used them to implement LFO functionality in the Hexter application. As a result of this work, Hexter now has all of the features required to perform *Madonna of Winter and Spring*, although the software itself it is still under development. The eventual aim is to provide functionality that is equivalent to that provided by the original DX-based synthesizers, so that a diverse range of works can be supported by the software.

2.5. Testing the system

In order to test the PD patch for *Madonna of Winter and Spring*, we created a number of MIDI files containing excerpts from the score. A number of short (5–10 bar) phrases were selected based on their use of control change and other real-time parameter changes. We converted patch data for the piece from Galaxy librarian bundles (provided by the composer) to DX7 System Exclusive format using Galaxy librarian to perform a SysEx dump to a laptop. The MIDI files along with audio renderings created using Hexter are available from the project weblog. The sound quality of the recordings is very good ('cleaner' sounding than the original instruments), and it is worth noting that the sequences ran smoothly without introducing significant load in PD. A maximum CPU load of 30% was observed on a 2.0 GHz Centrino-based laptop; our recommended specification for performing the work is a Dual processor G5 Apple Macintosh.

2.6. Building a user interface

Once the DSSI hosting and synthesis aspects of the project had been tested successfully, we began building a control interface in PD. For efficiency, one instance of 'dssi~' was used to host all ten instances of Hexter, with instances 1–8 emulating the TX816, and instances 9–10 emulating the DX1. Appropriate MIDI mapping was set up so that all program changes and performance parameter changes could be controlled remotely by the performers. We also developed a simple PD-based GUI for controlling various mappings and performance parameters for each instrument. Patch design was kept as generic and modular as possible so that patch elements could be reused for other works.

2.7. Packaging for Mac OS

Although we developed the system using the Linux platform, the target platform for this particular work, and possibly for other works, will be Mac OS X. In order to make the system as user friendly and easy to install as possible, we decided that the Mac OS PD-extended application developed by Hans-Christoph Steiner (available from http://at.or.at/hans/pd/installers.html) would be adopted. In order to reduce download size, all non-essential elements (externals, libraries and documentation) were deleted from the bundle. A Mac OS '.dmg' file was then created containing binary versions of Hexter and all of its dependencies including liblo, libdssialsacompat and GTK+. The installer for *Madonna of Winter and Spring* is less than 10 MB in size and is freely available for download from the project weblog site. It has been tested and found to work on Mac OS 10.3, and an OS 10.4 installer is currently being developed.

2.8. Testing

In late March 2006, we completed a prototype production system for *Madonna of Winter and Spring*, and proceeded to test it in collaboration with the composer. The composer expressed approval with regard to the overall approach taken towards porting this and other works, and in general seemed to be pleased with the responsiveness and accuracy of the system. He noted a number of specific issues that required improvement or modification. These included changes to controller and internal mappings, and some discrepancies in LFO detail. His comments have been recorded on the project weblog, and are currently being addressed.

Another outcome of the testing process was an observation that although the system performed well throughout the piece, in the most demanding sections a little more 'overhead' would be desirable. This would be in order to prevent audible signal break-up due to polyphony saturation through pedal use, and other DSP-intensive processes. We are currently investigating the possibility of utilising parallel processing to solve these problems.

3. THE INTEGRA PROJECT

Integra (www.integralive.org) is a three-year project led by UCE Birmingham Conservatoire in the UK and part financed by Culture 2000. The purpose of the Integra project is to develop a new environment for making music with live electronics, and to modernise a range of works that use old technology. The porting of *Madonna of Winter and Spring* has therefore become part of the Integra project.

As part of the first phase of development of the Integra environment, an OSC namespace is being created for passing performance data between control and processing modules in an implementation-neutral manner. The rationale behind this is that implementations of DSP elements (synthesis, processing, control) can change as new hardware and software is developed, but the meta-data describing how these elements interconnect and react during a performance stays constant. Once a suitable namespace has been developed, it should be possible to store all of the data required to perform a particular work in a platform- and environment-neutral format. A prerequisite of performing a work would then be that the relevant namespace-compliant modules must be available. Therefore, the first phase of development will involve the porting of thirty existing works using the Integra model.

The implication of this for the system developed for *Madonna of Winter and Spring* is that the DX1 and TX816 abstracted modules will be re-implemented as Integra modules. This will entail hosting the DX1 and TX816 with separate 'dssi~' instances and creating OSC wrappers for each instance (Figure 4). The DX1 and TX816 will therefore be presented to the user as two modules that are controlled entirely by Integra-compliant OSC messages. The modules will be conceived as part of the Integra module class hierarchy. For example, a TX816 emulator could inherit attributes from the 'Yamaha DX synth' class, which in turn would inherit attributes from the 'FM Synth' class, which in turn could inherit from 'Synth', and 'Audio Object', etc. The specification for this class hierarchy is currently under development.

Initially the modules for DX series synthesis will be implemented in Pure Data using 'dssi~' and Hexter, but the eventual aim is to port them to a range of other environments (such as Supercollider, Max/MSP and CSound) that support dynamic loading of libraries. CSound already has a DSSI host implementation, which is under development, and it would be relatively straightforward to port 'dssi~' to Max/MSP.

One of the goals of the Integra project is to provide a framework for storing data about technical elements of musical works involving live electronics, and to provide mechanisms for using those data to perform pieces. The framework should provide tools for exporting performance data from an environment, storing it, and re-importing it. It should also provide an environment-neutral graphical front-end for visualising Integra modules and designing performance systems. Since it will only output Integra-compliant OSC data, this GUI could be used with any DSP environment that provided the relevant Integra OSC-enabled modules.

4. POTENTIAL PROBLEMS

When a legacy work is realised with modern equipment, there is usually a noticeable 'improvement' in sound quality. This change in sound quality is mostly the result of higher sampling rates and higher bit depths. It is also related to improvements in filtering, dither algorithms, and other advanced DSP techniques. However, if the aim of porting the work is to remain perceptually faithful to the sound of the equipment it was written for, then 'improved' sound quality may not be desirable. One of the current ideas for consideration by Integra is to provide facilities for enabling different renderings of the live electronics for works, depending on the sound quality desired. For example, if the sampler or synthesizer originally used for a piece used 8-bit 22

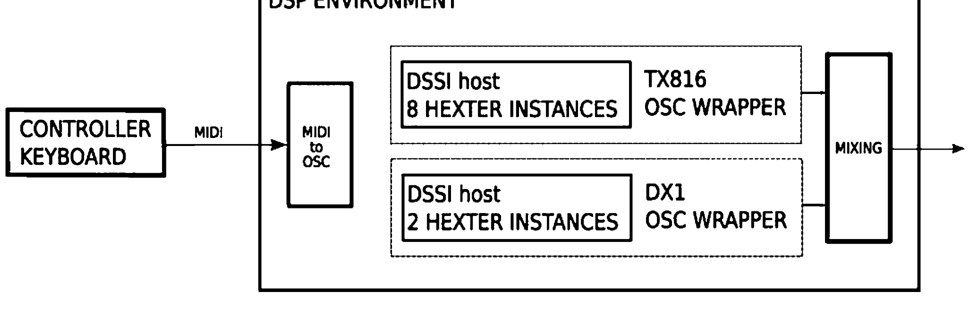

Figure 4. Proposed Integra DX synthesizer patch.

kHz DSP, then there should be an option in its emulation module to operate with these parameters. It would then be up to the performers and/or composer of the work to decide whether this was the preferred option.

Another issue with developing the system proposed in this article is that although the software and systems created will be free to the end user, and released under licences that guarantee the freedom to modify and/or distribute the software, it doesn't mean the software is without cost. Time taken to develop systems for performing works with live electronics, and to migrate repertoire to these systems must ultimately be funded. It is therefore important to consider cost implications even when developing 'free' software, and to carefully manage the way in which development time is utilised.

The emphasis of the Integra project so far has been on developing tools that can be used as building blocks in more specific systems. It is the purpose of the Integra project to continue to develop such tools in a co-ordinated and modular way. However, developing tools and releasing them under an open source licence doesn't guarantee their longevity or success. The aim of the migration and preservation work of the Integra project is to conduct porting in such a way that future efforts to preserve the works are minimised. In order to achieve this it is essential that all migration efforts are thoroughly and clearly documented, and that documentation is stored and presented in an efficient and accessible manner. It is also a project objective that an ethos of openness and transparency is adopted in order to encourage participation.

4.1. Implications for composers and publishers

In order for the repertoire migration efforts of the Integra project to truly succeed, and pervade beyond the three-year initial lifespan of the project, migration and preservation issues must be considered by both composers and publishers. In his survey, 'Record keeping practices of composers', Longton (2004) concludes that out of 161 composers questioned, 'almost half (47 percent) … lost files they considered valuable due to hardware or software obsolescence'. He continues to conclude that the lack of concern for preservation (of their work) indicated by responses to his questionnaire reflects the fact that 'composers have not historically had to concern themselves with preservation'.

This is corroborated by Polfreman, Sheppard and Dearden (2005) who note that 'publishers and composers rarely automatically update their archives and transfer to new media', but instead adopt a reactive approach 'with demand for a performance leading to updates'.

It is the view of the current authors that both composers and publishers need to take a more proactive approach towards preserving musical works involving technology. For composers this means adopting open standards where possible, and providing sufficient documentation for their works to be realised with alternative resources. For publishers this entails providing the necessary investment and resources to support the modernisation of existing works, as well as engaging with the efforts of projects like Integra. The Integra project seeks to greatly simplify these processes by providing a common technical and semantic framework for dealing with works involving live electronics.

5. CONCLUSION

The modernisation of live electronics in two works by Jonathan Harvey has been discussed in the broader context of the Integra project. A generic open-source solution for emulating Yamaha DX-based synthesis in contemporary works has been proposed. The approach suggested involves the use of DSSI plug-ins within the context of multimedia programming environments (such as Pure Data and Max/MSP). With its various components being licensed under open-source licences, this system has the advantage that new features can be added at any point if required. Although there is a development cost, the system entails no software cost to the end user or performer wishing to use it. Although the amount of development time taken to create the tools for performing works involving DX-based synthesis has been significantly greater than it would have been using solutions involving commercial tools such as Native Instruments FM7, it is hoped that this effort will greatly reduce the time required to modernise other works that use similar techniques. There are now a further three works (by Gérard Grisey, Philippe Hurel and Marc-André Dalbavie) that will be ported before September 2006, and it is hoped that this porting effort will lead to many more performances of these works. The migrations will be conducted using Integra modules created by making OSC wrappers for the Hexter FM DSSI plug-in hosted by the 'dssi~' PD object. Modernising these additional works will provide a test for the system's robustness, and highlight areas requiring further development.

REFERENCES

Arumi, P., and Amatriain, X. 2005. CLAM, and object oriented framework for audio and music. *Proc. of the Linux Audio Conf.*, pp. 43–50. Karsruhe, Germany: LAD.

Longton, M. 2004. Record keeping practices of composers. http://www.interpares.org

Phillips, D. 2005. Where are we going and why aren't we there yet? *Proc. of the Linux Audio Conf.*, pp. 147–9. Karsruhe, Germany: LAD.

Polfreman, R., Sheppard, D., and Dearden, I. 2005. Re-wired: reworking 20th century live electronics for today. *Proc. of the Int. Computer Music Conf.*, pp. 41–4. Barcelona: ICMA.

Puckette, M. 2004. New public-domain realizations of standard pieces for instruments and live electronics. *Proc. of the Int. Computer Music Conf.* Miami, USA.