Establishing connectivity between the existing networked music notation packages *Quintet.net*, *Decibel ScorePlayer* and *MaxScore*

Stuart James, Cat Hope, Lindsay Vickery, Aaron Wyatt
Western Australian Academy of Performing Arts, Edith Cowan University; Sir Zelman Cowen School of Music, Monash University
{s.james,l.vickery}@ecu.edu.au, cat.hope@monash.edu, music@psi-borg.org

Ben Carey, Xiao Fu, Georg Hajdu
Center for Microtonal Music and Multimedia (ZM4)
Hamburg University of Music and Theater (HfMT)
{ben.carey,xiao.fu,georg.hajdu}@hfmt-hamburg.de

ABSTRACT

In this paper we outline a collaboration where live internet-based and local collaboration between research groups/musicians from *Decibel New Music Ensemble*[1] (Perth, Australia) and *ZM4*[2] (Hamburg, Germany), was facilitated by novel innovations in customised software solutions employed by both groups. The exchange was funded by the *Deutscher Akademischer Austauschdienst*[3] and *Universities Australia*[4]. Both groups were previously engaged in the research and performance of similar musical repertoire such as John Cage's 'Five' (1988) and 'Variations I-VIII' (1958-67) among others, the performances of which utilise graphic, animated and extended traditional Western music notation. Preliminary steps were taken to achieve communication between the three existing network music notation packages, the *Decibel ScorePlayer*, *MaxScore* and *Quintet.net,* facilitating a merging – and ultimately an extension – of notational approaches previously prescribed by each music notation package. In addition to the technical innovations required to achieve such a project, we consider the outcomes and future directions of the project, as well as their relevance for the wider contemporary music community.

[1] Retrieved 26th Nov 2016 from http://www.decibelnewmusic.com/
[2] Retrieved 26th Nov 2016 from http://www.hfmt-hamburg.de/forschung/zm4/
[3] Retrieved 26th Nov 2016 from https://www.daad.de/en/
[4] Retrieved 26th Nov 2016 from https://www.universitiesaustralia.edu.au/

1. INTRODUCTION

Although many existing software solutions allow the display of music notation outside the conventions of traditional Western music notation – graphic, animated, extended traditional Western, and real-time notation (refer to Table 1) – a standardised format for communication between these software packages in real-time is yet to be established. Open Sound Control[5] and MIDI have long offered increasingly comprehensive solutions for dealing with sound generation, but have a few shortcomings when it comes to dealing with the transmission of rich graphics required for real-time eScore delivery in live performance [1]. This leaves performers of such music with the choice of only performing music using the prescribed methods employed by existing packages, or innovating their own software solutions. Performers seeking to collaborate with other innovators in the same field face further limitations in that these software solutions are likely to be incompatible due to their software design, communication protocol, and the command syntax adopted.

In order to overcome this problem, the strengths and limitations of the various software packages used by the Decibel New Music Ensemble and the ZM4 research group (respectively the *Decibel ScorePlayer* [2] App[6] for the Apple iPad, and *Quintet.net*[7] [3] and *MaxScore*[8] [4]

[5] Commonly known as OSC. Refer to Wright, M., Freed, A., & Momeni, A. (2003). Open Sound Control: State of the Art 2003.' International Conference on *New Interfaces for Musical Expression*, Montreal, p. 153-159.
[6] Retrieved 26th Nov 2016 from https://itunes.apple.com/au/app/decibel-scoreplayer/id622591851?mt=8
[7] Retrieved 26th Nov 2016 from https://quintetnet.hfmt-hamburg.de/wiki/pages/w7u7v9j3/Download.html

with its recent extension, the *NetCanvas* [5], for computers running MacOS or Windows) were documented (see Table 1) alongside other real-time notation packages to reach a viable standard for inter-application communication, in the fashion of previous efforts such as the *Max-Unity3D Interoperability Toolkit* [6], which utilises TCP/IP socket connections to transfer messages in real-time between *Max* and *Unity3D*, and *PWGL* and *Noteability* which instead support the OSC standard [7].

Some of the primary questions raised in such a collaboration are largely driven by how it may be possible to overcome some of the intrinsic differences of music notation packages in such a way that it does not mean re-inventing the wheel. There are a number of facets of this consolidation that are worth considering: the type of score and the scope[10] of the notation used, whether it is fixed or animated, the nature of networked performance and whether it is intended for both local area network (LAN) and wider area network (WAN) performance situa-

Software	Connectivity	Protocol and Data Type	Traditional Notation	Graphic Notation
Quintet.net (1999)	• Quintet.net Client • Quintet.net Server • Quintet.net Conductor • Quintet.net Viewer	Max messaging via UDP and TCP socket connections	Both Generative and Fixed via MaxScore	Yes. Fixed Animated, Live-Animated, and Live-Generative via MaxScore (refer to Figure 1) [8]
Bach [9] (2012)	No native support	Messaging is possible using Max over UDP or TCP	Both Generative and Fixed	via slot messages, requiring an external drawing object
InScore [10] (2012)	No client / Server architecture per se, rather the InScore application responds to remote OSC messages received	OSC Packets over UDP	Both Generative and Fixed	Yes. Fixed Animated, Live-Permutated, Live-Animated, and Live-Generative
Decibel ScorePlayer (2013)	Decibel ScorePlayer Client / Server over a TCP socket connection	OSC Packets over UDP	Limited	Yes. Fixed Animated, Live-Permutated, and Live-Animated
MaxScore.NetCanvas (Apr-Nov 2016)	• Internet browser application • NetCanvas Server • NetCanvas Maxpatch abstraction • NetCanvas virtual-reality / browser client	Rendered PNGs are streamed as formatted packets of base-64[9] byte arrays over a TCP web-socket connection	Both Generative and Fixed	Yes. Live-Animated

Table 1. Comparison of connectivity, protocol, and notation types for existing ScreenScore applications.

[8] Retrieved 26th Nov 2016 from http://www.computermusicnotation.com/

[9] 64-bit

[10] Arguably, in the case of some animated scores, constraints may be necessary, and such solutions may not address the scope of what is possible with openGL and VR notation. The game industry, for example, has already had to define a framework and syntax for animation.

tions, and the technical aspects of integrating both the networking protocol and syntax to allow such software solutions to talk with one other and remain synchronised. These technical aspects follow some key central ideas: the interoperability of digital score solutions, multiple devices, and software platforms; the need to evaluate existing software dependencies of digital score solutions (see Table 2); the need to further extend the feature set of such solutions; and the need to consolidate some of the intrinsic differences between existing software solutions. Such aims of interoperability investigated include:

- Integration and synchronisation of Quintet.net and the Decibel ScorePlayer
- Development of a Canvas score module for the Decibel ScorePlayer and a Max abstraction responsible for converting the draw commands from MaxScore to the Canvas score in the ScorePlayer
- Transmission of OSC via network sockets (over TCP/IP or UDP) to the MaxScore.NetCanvas Server from the Decibel ScorePlayer and Max 7
- Integration of support for OSCblob message packets over UDP to the ScorePlayer app from Max 7, decoded as PNG textures in the Decibel ScorePlayer
- Integration of support for layer creation/deletion, cursor automation, via existing Decibel message syntax in MaxScore.NetCanvas from Max 7

Music Notation Package	**Software Dependencies**
Quintet.net	Max and various 3rd party externals packages
MaxScore	Max, Java Music Specification Language (JMSL), and 3rd party externals packages
Bach	Max
InScore	An external application capable of sending OSC messages
Decibel ScorePlayer	None
MaxScore.NetCanvas	Max, Java Music Specification Language (JMSL), and 3rd party externals packages

Table 2. Comparison of Music notation Packages and their software dependencies.

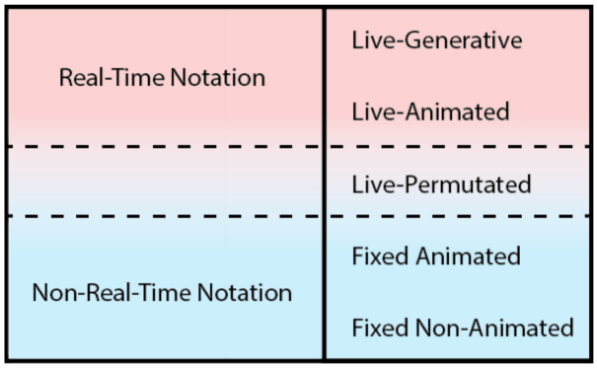


Figure 1. Categories of real-time and non-real-time music notation. (Shafer, 2016).

2. DEVELOPMENTS IN THE DECIBEL SCOREPLAYER

When the Decibel ScorePlayer was first developed, it was conceived of as a standalone application without intrinsic software dependencies. The application specifically focussed on graphic and animated notation on a portable device (the iPad), with the intention that multiple devices could be synchronized in real-time over a Wi-Fi network. The Decibel ScorePlayer is hard-coded in objective-C and the binary application currently supports the iPad generation 2 and 3. The motivations in using such an application were driven largely toward ease of use for the end-user, both composer and performer.

The ScorePlayer adopted the zero configuration networking DNS service discovery [11] method in order to streamline the process of connecting multiple devices over the network. Each device broadcasts a service name determined by the score that is open, and the ScorePlayer application adopts its own client-server method involving the management of both a primary and secondary server and remaining clients on the network. The secondary server is used as a backup primary server in case the existing primary server drops off the network. This networking architecture has proven to be robust in live performance settings. Over a period of some years of using this in live performance, Decibel New Music ensemble have found this to be a robust and reliable solution for synchronously (or asynchronously) presenting scores over multiple devices.

A number of enhancements to the Decibel ScorePlayer were completed as part of the *Deutscher Akademischer Austauschdienst* and

Universities Australia exchange. These included providing WAN capability for the ScorePlayer application, and the completion of several modules for the ScorePlayer allowing the application to function in a number of different modes: a 2D scrolling "Talking Board" mode, a generative notation "Rodinia" Conductor/Performer/Audience mode, and a blank externally controllable "Canvas" mode.

2.1 WAN Internet Connectivity

The ScorePlayer was originally intended for synchronising multiple iPads over a Local Area Network (LAN) [12]. However, one of the first proposed outcomes of the research exchange was to enable the ScorePlayer to synchronise over a Wide Area Network (WAN) for telematic performance. Decibel ensemble member Aaron Wyatt extended the ScorePlayer to facilitate this process, allowing the user to manually enter a destination IP address so that a local iPad, assuming the role of a client via TCP, can remotely connect to another iPad which in turn assumes the role of a server (refer to Figure 3). This development, after some testing, was incorporated as part of a telematic concert on occasion of the Sound and Music Computing Summer School 2016 between the *Hochschule für Musik und Theater* in Hamburg, Germany, *Edith Cowan University* in Perth, Western Australia, and *Stanford University* in California, United States. The concert successfully presented a series of new works by participants of the Summer School featuring performers from three different countries around the world (**Figure 2**).

Figure 2. Three-way connectivity in a telematic performance employing JackTrip for audio streaming and the Decibel Score Player for synchronized score rendering across 15 time zones.

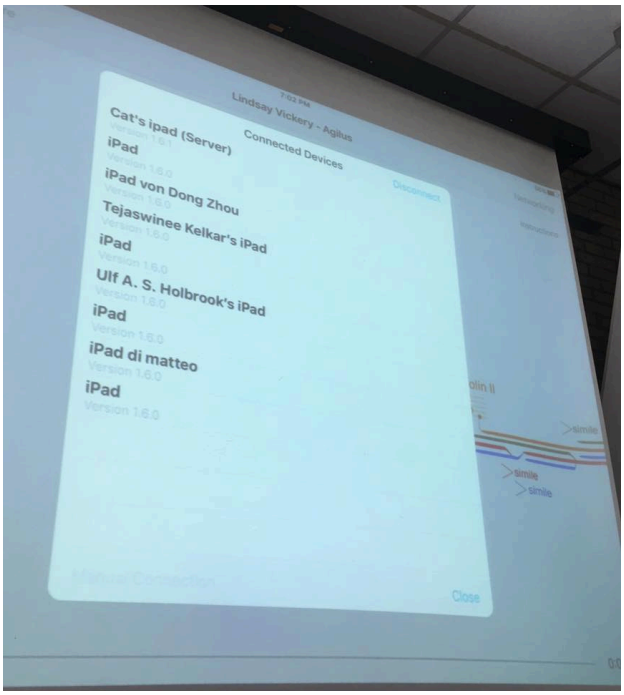


Figure 3. The iPads connected during the telematic performance at SMC2016 in Hamburg.

2.2 Talking Board Mode

In Vickery and Hope's work 'The Talking Board' (2011) [12] a graphical score-collage is continuously repositioned during the performance, moving smoothly in the vertical and horizontal dimensions, and also jumping to particular new positions. The four performers realize the work by interpreting the components of the score that are framed by four colour-coded planchettes (circles). Rather than following defined pathways, the planchettes move in 2-dimensions according to a set of behaviours (wander, converge, flock, lead). An OSC networking protocol was implemented in the ScorePlayer to synchronise the iPad score with an external Max patch [13]. In the "talking board" mode, external communication was introduced to allow data on the score and planchette positions to be used to control spatialisation and processing within Max.

The behaviours are transmitted via OSC using the following messages:

/External/MoveBackground x1 y1
(the xy position of the background collage).
/External/MovePlanchette x1 y1
(the xy positions of each planchette).

2.3 Rodinia Mode

Rodinia is a varied paradigm in the ScorePlayer: a networked, generative and interactive, conducting environment [14] allowing for control by four "conductors" of generative notation for four ensembles. The score includes three view paradigms: audience, conductor and performer. The work draws in part from the generative functions developed for the *Decibel Cage Variations App* (Variations I and II).

Vickery's work 'TECTONIC: Rodinia' (2016) employs a collision avoidant algorithm which may modify the choices of each conductor. As notational streams approach one another they are pushed upward or downward according to their evaluated mass. The controller interface is operated by two hands (the iPad permits 11 simultaneous multi-touch points) [15] allowing parameters to be specified simultaneously by the Left hand (play/hold, articulation, duration type) and Right hand (duration, pitch, dynamic, rate and compass) (see Figure 4).

Figure 4. "Conductor View" in Lindsay Vickery's 'TECTONIC: Rodinia' (2016).

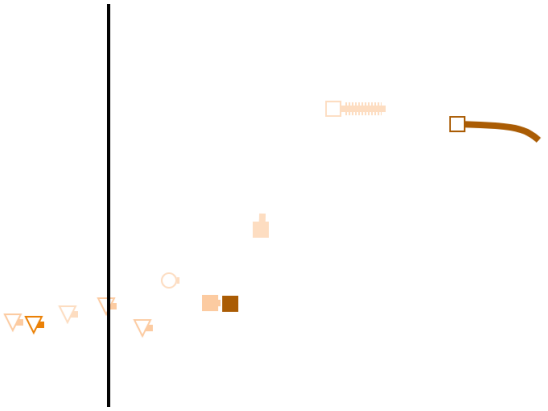


Figure 5. "Performer View" in Lindsay Vickery's 'TECTONIC: Rodinia' (2016).

These parameters define the boundaries of stochastically generated graphical events which are distributed to the all of the iPads belonging to the same stream on the network. Musicians read the generative score in a "performer view" where notation for each of the four ensembles is scrolled right to left across the iPad screen (see Figure 5).

The "audience view" amalgamates the notation from each stream into a single score, to be shown on a large screen behind the performers for both audience and the conductors. Audience view draws the streams of notation approaching from four directions (left, right, top and bottom) (see Figure 6). The notation "wraps" around each time it completes the crossing from one side of the score to the other.

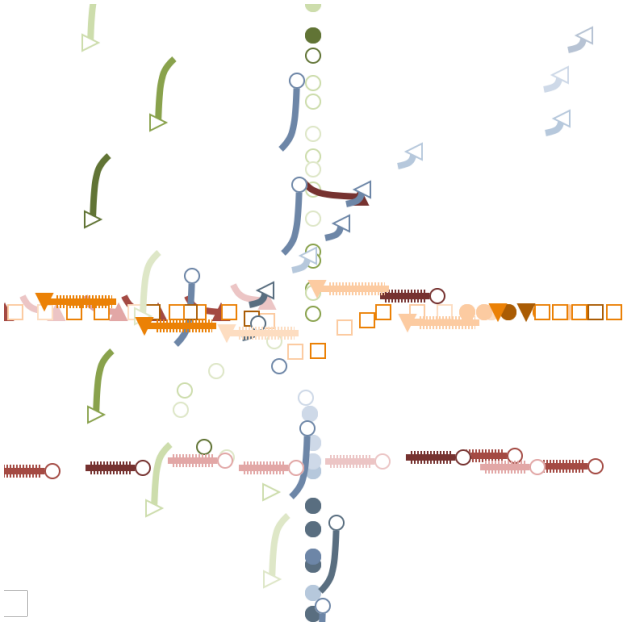


Figure 6. "Audience View" in Lindsay Vickery's 'TECTONIC: Rodinia' (2016).

The full range of conductor defined commands as well as the current xy positions of each part in the "audience view" are transmitted to Max as OSC commands:

/External/Hold – (allowing the generative of notation for a group to be halted).
/External/Articulation – (defining 6 articulation types)
/External/DurationType - defining 4 duration types)
/External/BarValue - (defining duration, pitch, dynamic, rate and compass)
/External/Event – (xy position of each group)
/External/PlayerNumbers (defining the number of players in each group)

This data is used to process and spatialise the sound of each instrumental group according to the conductor defined parameters in real-time.

2.4 Canvas Mode

The Canvas mode principally varies from previous modes in that it accepts external messages defining core elements of a digital score, such as the dynamic creation of layers and loading and positioning of image files. In Canvas mode the ordering, occurrence and motion of the score segments can be determined externally by OSC messages sent via UDP. Previously all score generation – as found in the iPad implementations of Cage 'Variations I' (1958) and 'Variations II' (1961) or Vickery's 'TECTONIC: Rodinia' (2016) – and indeterminate elements – as found in Hope's 'Liminum' (2012) or Vickery and Rose's 'Ubahn c1985: the Rosenburg Variations' (2012) – were generated using randomised procedures within the ScorePlayer application.

Canvas mode allows composers to implement generative and indeterminate works in the ScorePlayer without the generative procedures having to be hard-coded into the ScorePlayer application. The concept was trialled with a pre-existing score: Samuel Dunscombe's 'Westpark' (2012) for bass flute, bass clarinet and electronics.[11] In the original version of the work, the score comprised 46 indeterminately presented images and 4 dynamic markings that were screened on networked laptops. Dunscombe intends the performers to react almost instantaneously to the often rapidly changing images.

The Canvas mode was used to define a background score layout and to place the score images and dynamics within separated parts for each instrument.

A Canvas mode score is defined by an .xml file formatted as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE opus SYSTEM "opus.dtd">
<opus>
   <score>
   <name>Title</name>
   <composer>Composer Name</composer>
   <type>Canvas</type>
   <duration>0</duration>
   <instructions>workinstructions.png</instructions>
        </score>
</opus>
```

This xml file and the required image files are compressed in a .zip file, which is then given the extension .dsz and loaded into the Decibel ScorePlayer via iTunes[12]. Messages to define the score changes from Max are transmitted as OSC packets to the ScorePlayer over UDP. In the Decibel ScorePlayer, a single image must be loaded into a discrete graphics layer. The canvas mode score accepts messages in the following formats:

/Renderer/Create "name_of_layer" n x1 y1 x2 y2
(Where n defines 0 a layer that appears in all parts or 1…n the part in which this layer appears. x1 and y1 define the image (top left) x and y position and x2 and y2 define the image (bottom right) x and y position in pixels.)

/Renderer/LoadImage "name_of_layer"
imagename.png n
(where n is either 1 to auto size the image/0 to use the image's "actual size")

/Renderer/Remove "name_of_layer"

/Renderer/SetPosition "name_of_layer" x y
(Where x and y define the top left coordinate of the image in pixels).

Tests of the Canvas mode version (see Figure 7) of the score suggest that the iPad score load images more quickly than the original Max version even though the commands are being transmitted across a network.

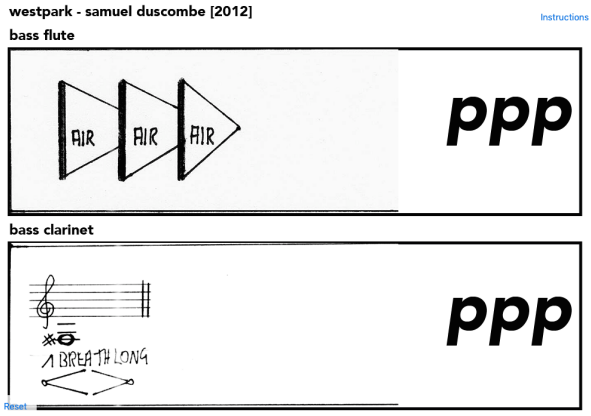


Figure 7. Decibel ScorePlayer version of Samuel Dunscombe's 'Westpark' (2012)

[11] It also was used to create a new version of Vickery's work 'abstract clouds of the western skies' *(2016)* which had been originally authored in MaxScore as part of the exchange.

[12] Retrieved 26th Nov 2016 from https://vimeo.com/140709500

The Canvas mode interestingly permits a greater degree of "animation" than is normally found in the Decibel ScorePlayer as layers can be repositioned and resized independently. This possibility has not yet been exploited in a creative work.

2.5 Interoperability between MaxScore and the Decibel ScorePlayer

One of the further developments in progress as a result of this research exchange has been the investigation as to the viability for MaxScore to use the Decibel ScorePlayer as a canvas window to display generative notation. This process has required sifting through the draw commands Nick Didkovsky has formalised in [mxj com.algomusic.max.MaxScore]. Currently, Georg Hajdu has implemented the MaxScore canvas in Max using JavaScript mgraphics commands. However, as the draw commands in the Decibel ScorePlayer Canvas module are different, an abstraction in Max is necessary and responsible for translating the commands from MaxScore into the appropriate syntax. Furthermore, the display ranges are normalised according to the size of the display window, and some adjustment is made depending on the anchor position of each graphical element displayed onscreen.

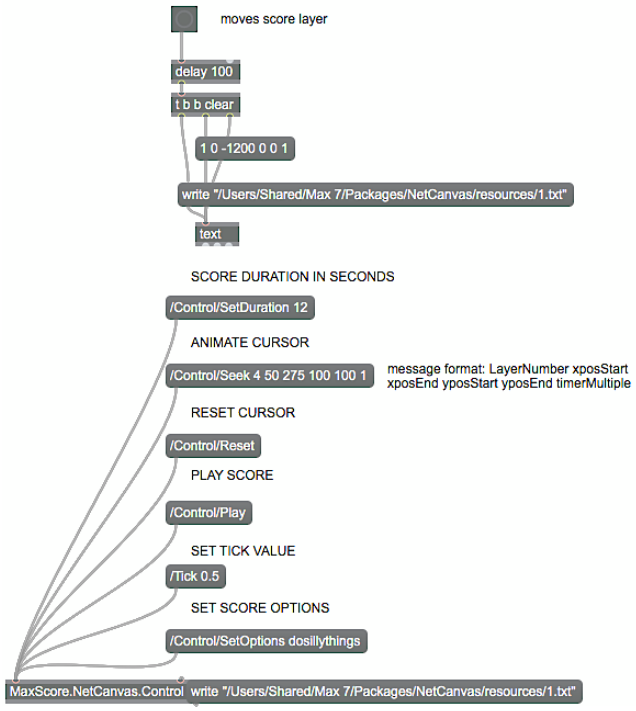
MaxScore	Decibel ScorePlayer
staffLine 0. 0. 0. 0.5 20. 51. 781.79364 51.	/Renderer/Create staffline1 0 24 70 949 60 /Renderer/LoadImage staffline1 StaffLine.png 0
tr 22. 75.96 0.5 Staff 0. 0.	/Renderer/Create trebleclef 0 26 161 176 311 /Renderer/LoadImage trebleclef TrebleClef.png 0

Table 3. Comparison of draw commands in MaxScore and the Decibel ScorePlayer.

The research exchange has also been concerned with the real-time transmission of graphical data over the network. Certain restrictions in the size of an OSC full-packet support the idea of splitting large files into smaller packets. Research is in progress to determine the best way to transmit media over the network.

3. DEVELOPMENTS IN MAXSCORE.NETCANVAS

As a result of this collaboration, and in line with development goals set out for the MaxScore.NetCanvas project, a number of changes were made in anticipation of the release of NetCanvas beta 0.2. Firstly, NetCanvas now supports the creation of multiple graphical layers, which can be animated via command sets that can be created in Max and sent via the NetCanvas Server. Parameters can be changed in real-time via messages that follow the ScorePlayer message syntax (see Appendix 1). This major change allows for scrolling scores, opacity between layers and more exotic animations of layers, which can be updated via websocket connections made in the browser. Secondly, cursor support was implemented following the logic of Georg Hajdu's composition 'Carnage', originally composed for a joint concert between the two research teams in Perth in July 2016. Cursors can be controlled independently, instantiated and destroyed, set to animate in a variety of states and set to different shapes, sizes and colours (Figure 8).



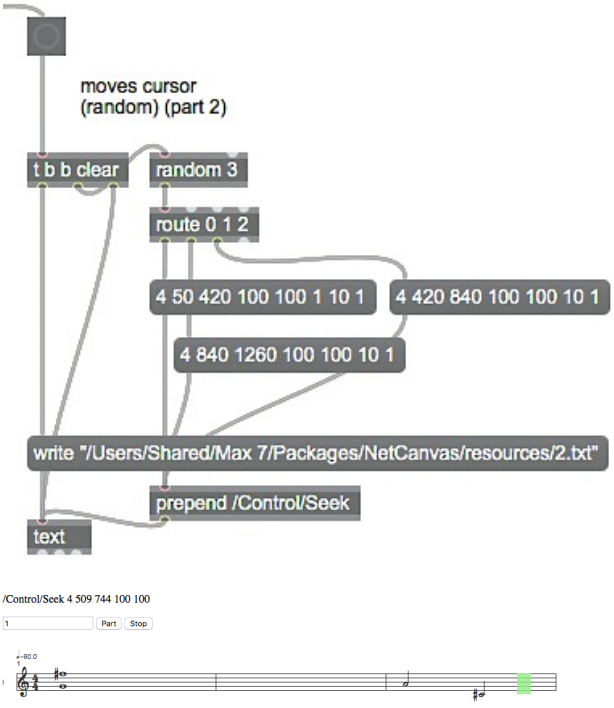


Figure 8. Sending customised cursor instructions from Max to the NetCanvas using the messaging format supported by the Decibel ScorePlayer.

```
final String textVar = ( UIMain.usersPath + "/" + sessionCounter + ".txt" );
    try {
        File j = new File(textVar);
        FileInputStream bt = new FileInputStream(j);
        ByteArrayOutputStream tout = new ByteArrayOutputStream();
        IO.copy(bt, tout);
            ByteBuffer byteBuffer = ByteBuffer.wrap(tout.toByteArray());
            mSession.getRemote().sendBytes(byteBuffer);
            tout.close();
            System.out.println( byteBuffer);
            byteBuffer.clear();
            }
            catch (IOException e) {
                e.printStackTrace();
            }
```

```
function curs1(x) {
// Score duration
x = x.split(" ");
if (x[0] == "/Control/SetDuration") {
    var duration = x[1];
    document.getElementById("message").innerHTML = "Score duration is set to " + duration;
    }

else if (x[0] == "/Control/SetOptions") {
    var options = x[1];
    document.getElementById("message").innerHTML = "Score options are set to  " + options;
    }

else if (x[0] == "/Control/Play") {
    var options = x[1];
    document.getElementById("message").innerHTML = "Play Started";
    }
else if (x[0] == "/Control/Reset") {
    var options = x[1];
    document.getElementById("message").innerHTML = "Score reset to initial state";
    }
else if (x[0] == "/Tick") {
  var tick = x[1];
 document.getElementById("message").innerHTML = "Current Tick value is " + tick;
}
// Seek commands for layers
else if (x[0] == "/Control/Seek") {
document.getElementById("message").innerHTML = (x[0]+ " " + x[1] + " " + x[2] + " " + x[3] + " " + x[4] + " " + x[5])
  var lay = "layer" + x[1];
  var elem = document.getElementById("layer" + x[1]);
  var xpos = x[2];
  var ypos = x[4];
  var id = setInterval(frame, x[6]);
  var id2 = setInterval(frame, x[6]);
function frame() {
    if (xpos == x[3]) {
      clearInterval(id);
    } else {
      xpos++;
      elem.style.left = xpos + 'px';
    }
    if (ypos == x[5]) {
      clearInterval(id2);
    } else {
      ypos++
      elem.style.top = ypos + 'px';
    }
    }
  }
}
```

Figure 9. Changes made to the NetCanvas Server (above) and the NetCanvas (below) to support the new messaging format and cursor animation.

This involved some changes to the server code, and the NetCanvas code (Figure 9). Following the logic of the existing server implementation, users build text files containing animation instructions in Max, which are transmitted whenever any changes are made. This technique takes advantage of NetCanvas' use of websockets to push data to the browser clients, without requiring interaction on the client side. This makes for a very scalable system, which now supports real-time notation, animated notation, tablet and smartphone scores, network music performance, VR notation, and is fully cross-platform and accessible from Max, and can be used in conjunction with the Decibel ScorePlayer.

4. CREATIVE WORK

The teams form Hamburg and Perth completed and performed a number of works exploring screenscore-based notation (Table 4.). These works included the investigation of the possibilities of both MaxScore and the Decibel ScorePlayer**,** as well as one work exploiting the possibilities of both platforms.

New Works	
Carey, B.	*Magnetic Visions VI* (2016)
Fu, X.	*I Love Tiffany* (2016)
Hajdu, G.	*Carnage* (2016)
Hope, C.	*Great White* (2016)
Vickery, L.	*abstract clouds of the western skies (2016)*
Vickery, L.	*TECTONIC: Rodinia* (2016)
Adapted works	
Dunscombe, S.	*Westpark* (2012)
Vickery, L and Hope, C.	*Talking Board* (2012)

Table 4. Works developed during the residency.

Vickery's 'abstract clouds of the western skies' (2016) is a nonlinear work in which a 68 measure "source score" is deconstructed into passages and single measures to create a texture that alternates indeterminate juxtapositions with more synchronous linear "composed" passages. It is the second work by the composer in this formal framework (the first was 'Improbable Games' (2010)). During the exchange version for both MaxScore and the Decibel ScorePlayer

were created. The Canvas mode is used to define the score in the ScorePlayer, but unlike Dunscombe's 'Westpark' separate "parts" are defined for each of the three performers (see Figure 10). In the Decibel ScorePlayer performers change parts using a swipe up or down gesture on the iPad screen [11].

Figure 10. Flute part of Vickery's 'abstract clouds of the western skies' (2016) in MaxScore (above) the Decibel ScorePlayer (below).

The main difference between Quintet.net and the Decibel Score Player lies in how the progress of time is represented. While the ScorePlayer, in its basic mode, moves the score under a static play head, scores in Quintet.net are shown as static pages over which a cursor is moved from left to right. One of the pieces, that originally existed for Quintet.net and therefore had to be adapted to the ScorePlayer's paradigm is Xiao Fu's composition 'Tiffany' written in 2011 for a flexible ensemble of musicians and first performed as part of the Pentalocus network concert between San Diego, Montreal, New York, Belfast and Hamburg in November 2011. Her program notes state:

> *Works inspired by the Tiffany glass. I used the five logos of the participating school and split them into five parts, each part is processed, and then rearranged, in order to finally generate a colorful "Tiffany glass Score".*
> *The work's full-length is five minutes, each page is one minute. From left to right is time, from top to bottom is the approximate pitch range.*
> *If there is only one musical instrument in a given location, different colors represent different playing techniques; or if there is a duo or trio, different colors represent a different instrument.*
> *Each grid block can be a note, but the players are free to choose among simultaneous blocks and can even jump between them as long as they have the same color. White spaces are silence.*
> *The height of each small block, represents dynamics, the higher the box the louder.*

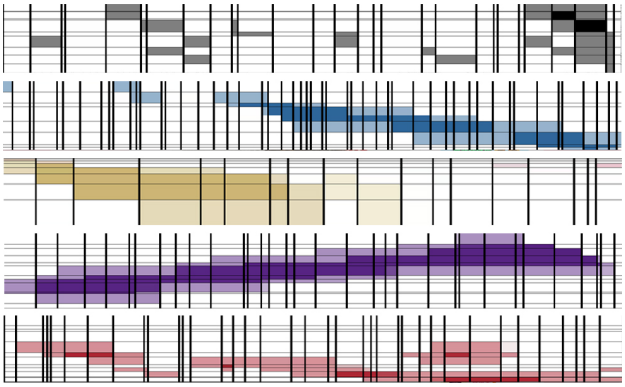


Figure 11. A page from Xiao Fu's composition 'I Love Tiffany' (2016) for a networked music ensemble.

The only work to explicitly explore communication between the Decibel ScorePlayer and Quintet.net at this point in the exchange is Cat Hope's 'Great White' (2016). This work was scored for accordion and viola (or for any two sustaining instruments) reading from the Decibel ScorePlayer and two laptop performers connected to Quintet.net.

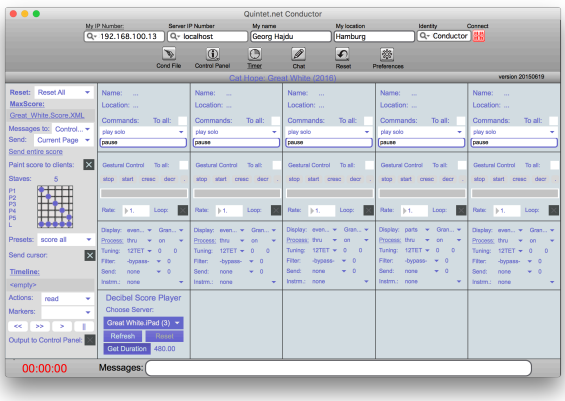


Figure 12. Screen shot of Quintet.net Conductor with built-in control of the Decibel Score Player.

Since the Quintet.net Conductor component—via an instance of the MaxScore Editor—serves the scores to the clients, we implemented a subpatch capable of communicating with the instance of the Decibel ScorePlayer acting as a server (see Figure 12). After a successful handshake, the Quintet.net Conductor starts the performance by sending the /Control/Play message to the server, upon which it receives a constant stream of ticks which it can sync its own clock to. In this way activity scored for Quintet.net was able to synchronise with the scrolling Decibel ScorePlayer score and therefore the acoustic instruments. This activity is indicated by four sections of traditional notations that appear in blocks on the score. The traditional notations are excerpts of works from composers who had spent some time in Hamburg: including György Ligeti, Gustav Mahler, Georg Telemann and Alfred Schnittke.

The performers of Quintet.net are provided with MIDI files of these excerpts, however certain notes have been removed by the composer beforehand. They are free to interpret these files in any way they wish for the performance, but can only do that when indicated in the score.

The title 'Great White' refers to the great white dead men of music history and women composer's struggle to find a place in that history. It is also a reference to the rare species of (occasionally "man-eating") shark that is hunted in the composers' home state of Western Australia. The acoustic instruments interface with the music excepts (already being altered by the electronic performers) firstly by reading the graphic score that 'sidesteps' the excerpt, then tracing along the next one, then increasingly 'interfering' with the remaining excerpts - eventually blocking out most of the excerpt altogether, concluding with general confusion and a 'free for all' to close. The Quintet.net performers perform with the midi files – try as they might, they cannot reproduce the masterworks as they are notated on the score that they follow.

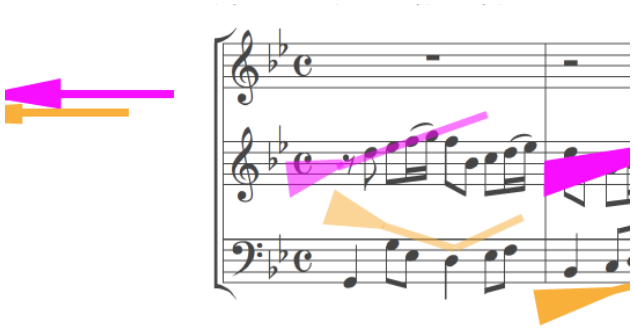


Figure 13. Score showing 'tracing', and some blocking, of the third music excerpt in Cat Hope's 'Great White'.

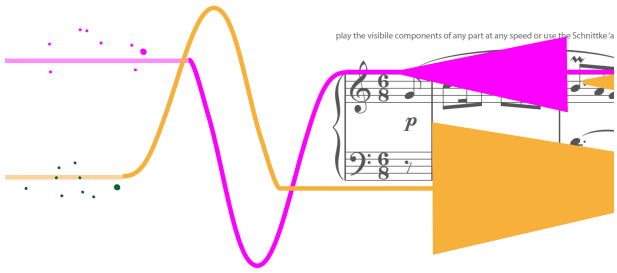


Figure 14. Score showing more 'blocking' of the parts.

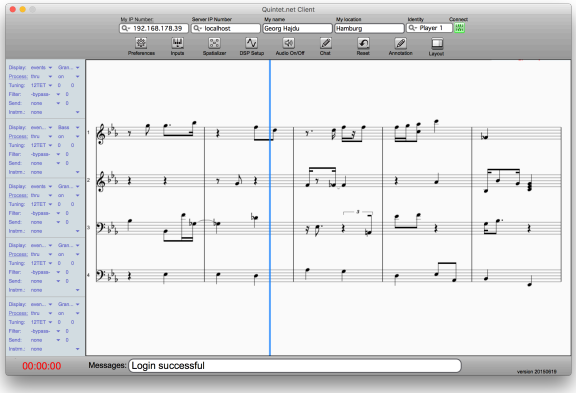


Figure 15. The version of the score for 'Great White' served by the Conductor to the Quintet.net Clients. The cursor is sync'ed to the ticks sent by the Decibel Score Player.

5. CONCLUSIONS

This research collaboration has allowed for some progress in allowing for interoperability between Quintet.net, the Decibel ScorePlayer, MaxScore. Future research will focus on extending the Canvas mode in the ScorePlayer to allow for drawing geometric shapes (and animated notation) using Core Graphics and Core Animation. It is hoped that through this collaborative research, some standardization can be established for communication between various digital score solutions, and that for the end user, there may be some flexibility in the way in which scores are delivered on screen. Future

development may also evaluate whether *Ableton Link* – which is fast becoming a de facto standard for synchronising devices to *Ableton Live* over local networks – is a worthwhile means of synchronising various digital score solutions. Although this approach, which has been a key feature of this collaboration thus far, is not designed to function over extended distances, a custom synchronisation method could also be developed or simply adapted from existing solutions.

Acknowledgments

The exchange between the Hochschule fur Musik und Theater (HfMT Hamburg) and Edith Cowan University (Perth) was funded by the Deutscher Akademischer Austauschdienst (DAAD) and Universities Australia.

6. REFERENCES

[1] J, Freeman. "Extreme Sight-Reading, Mediated Expression, and Audience Participation: Real-Time Music Notation in Live Performance," *Computer Music Journal*, 32(3), 2008, 25–41.

[2] C. Hope, A. Wyatt, & L. Vickery. "The Decibel Score Player – A Digital Tool for reading Graphic Notation." *Tenor Conference*, Paris, France, 2015.

[3] G. Hajdu, "Quintet.net: An environment for composing and performing music on the Internet," *Leonardo*, 38, 2005, 23-30.

[4] N. Didkovsky and G. Hajdu, "Maxscore: Music notation in Max/Msp," in *Proceedings of the International Computer Music Conference*, 2008, 483-486.

[5] B, Carey, & G, Hajdu. "Netscore: An image server/client package for transmitting notated music to browser and virtual reality interfaces." *Tenor Conference*, Cambridge, United Kingdom, 2016.

[6] I. I. Bukvic, & J-S Kim. "μ Max-Unity3D Interoperability Toolkit*." Proceedings of the 2009 International Computer Music Conference*, Montreal, Canada, 2009.

[7] C. Agon, G. Assayag, M. Laurson, and C. Rueda. "Computer Assisted. Composition at Ircam: PatchWork & OpenMusic." *Computer Music Journal* 23(5), 1999, 59-72.

[8] S. Shafer. "Performance Practice of Real-Time Notation." *Tenor Conference*. Cambridge, United Kingdom, 2016.

[9] A. Agostini, & D. Ghisi. "Gestures, Events, and Symbols in the Bach Environment." *Actes des Journées d'Informatique Musicale* (JIM 2012), Belgium, 2012.

[10] D. Fauber, Y. Orlarey, S. Letz, "INScore - An Environment for the Design of Live Music Scores", Proceedings of the Linux Audio Conference, LAC 2012.

[11] D. Steinberg. *Zero Configuration Networking: The Definitive Guide*. O'Reilly Media, 2005.

[12] A. Wyatt, C. Hope, L. Vickery, & S. James. "Animated Music Notation on the iPad," In *Proc. Int. Computer Music Conf. 2013*, Perth, Western Australia, 2013, 201-207.

[13] C. Hope, A. Wyatt, & L. Vickery. "The Decibel Scoreplayer: Enriched Scores for the iPad," in *Proc. Int. Computer Music Conf,* 2015.

[14] L. Vickery, & S. James. "Tectonic: a networked, generative and interactive, conducting environment for iPad," in *Proc. Int. Computer Music Conf. 2016*, Utrecht, The Netherlands, 2016, 542-547.

[15] K. Yarmosh. *App Savvy: Turning Ideas into iPad and iPhone Apps Customers Really Want.* O'Reilly Media, 2010, p. 53.

[16] L. Vickery, C. Hope, & S. James. "Digital adaptions of the scores for Cage Variations I, II and III," in *Proc. Int. Computer Music Conf. 2012,* Ljubljana, Slovenia, 2012, 426-432.

APPENDIX

List of Commands for the Decibel ScorePlayer.

Note: For externals, only the following classes of commands are forwarded to the iPad clients.

/Control /Renderer /Master
And only the following message classes are forwarded to externals.

/External (This is not broadcast to iPad clients.)

/Control /Status /Tick
The following commands are for client server interaction, and form part of the initial handshake.

/Server/ProtocolVersion ***versionString***
Sent by the server to the client on initial connection. The version string reflects the protocol version used by the server and is of the format "Decibel Networking Protocol v13".

/Server/RegisterDevice ***versionString identifier playerVersion***
Once the client receives the /Server/ProtocolVersion message, it needs to respond to the server with this message. The version string should match the one given by the server. If not, the player should abandon the connection and alert the user. The identifier is the hostname of the device. (Some rendering modules give you the option of specifying the identifier. This is mostly for interactions with externals.) The playerVersion is the version of the app itself. (At the time of this writing, the current development version was 1.6.4)

/Server/ConnectionOK
If the /Server/RegisterDevice command was successful, the server will notify the client with the above message.

/Server/MakeSecondary
Once the handshake is completed, the server sends this command to the client if there is not already a backup server on the network. The client then sets itself up as a secondary server and informs the primary server of the port number used for it.

/Server/SecondaryPort ***portNumber***
The return command sent upon the creation of a secondary server, including the listening TCP port number.

/Server/CancelSecondary
If the primary server doesn't get a response from the secondary server within a 3 second timeout period then it elects a new client as a secondary (if there are other clients on the network) and adds
the original secondary to a blacklist. It also sends this command to the failed secondary to let it know that its services are no longer required.

/Server/SecondaryServer ***address port***
Once a backup server has successfully been established, the primary server sends this message to all of the connected clients. The clients should then connect to the secondary server using the same handshake. If the primary server goes down then the secondary server seamlessly takes over.

/Server/RegisterExternal ***versionString udpPortNumber***
This command is used by an external (for example a Max patch) connecting to the server over UDP. The versionString should match the one used to establish a normal client/server connection, and the udpPortNumber should be a port that the external is listening on for return messages.

/Server/RegisteredExternals ***address1 port1 address2 port2...***
The /Server/RegisteredExternals message is sent from the primary server to the secondary server so that it can maintain an up to date list of externals in the case that it is called upon to replace the primary server. Multiple externals can be specified in the argument list.

/External/NewServer ***hostName udpPortNumber***
If the secondary server takes over, it sends this command to all of the registered externals to notify them that the server address has changed. The external should then re-register with the secondary server. (The secondary server clears the list of registered externals after sending this message, taking the server change as an opportunity to remove any dead externals from the list.)

/Server/GetClientList
Sent from a client to the server to get a list of all of the connected clients. (Used by the NetworkViewController to show the state of the network.)

/Server/ClientList ***identifier1 playerVersion1 identifier2 playerVersion2...***
After receiving a client list request, the server responds with this message. The arguments are a list of the identifiers (usually the hostnames) and the player versions of the connected devices.

/Malformed
This special message is sent internally within the server module to alert it to the presence of a malformed message within an OSC bundle. It is currently quietly ignored, as is the malformed message.
These next commands are more general control commands.

/Control/Play
Begins playback from the current location.

/Control/Reset
Resets the ScorePlayer. (Stops playback and sets the score back to the starting location.)

/Control/Seek ***location***
Seek to the specified location in the score. The location should be a floating point value between 0 and 1. For some scrolling scores, with an instruction area defined to the left of the starting point, this value can be set less than 0, but only while the score is stopped.

/Control/SetDuration ***duration***
Sets the duration of the score in seconds. This command should not be sent while the score is playing. (In newer versions of the score player it will be safely ignored if this is the case. In older versions, this could have unexpected results.)

/Control/SetOptions ***options...***
Sets options for the current score. The possible options, and the format the argument list should take, are defined by the individual renderer modules.

/Renderer
This class of message is passed by the player to the renderer class. The "Renderer" component of the address is added and stripped by the player to facilitate routing – the renderer itself has no knowledge of its existence.

/Status ***scoreName scoreType "Reserved" playerState location duration ("CurrentOptions" options...)***
Returned from the master in response to a /Master/GetStatus request. The scoreName and scoreType arguments are used to verify the name of the score and the rendering class used by it. (These were added in Protocol version 12 when manual network connections were first allowed. Zeroconf discovery only finds servers that are running the same score as defined by the name and composer.)
The "Reserved" argument is for the possible future implementation of score version checking. PlayerState shows the current state of the player – 0 for stopped, 1 for playing.
The location is the current location within the score as a float between 0 and 1, and the duration is the current length of the score in seconds. (Important for scores where the duration can be changed.)
The "Current Options" argument is optional, and is only present if the score has options that need to be set. The remaining arguments that follow are options and are dependent on the rendering module used.

/Tick location
The tick message is sent across the network every second by the master with a float argument between 0 and 1 which represents the current location within the score.

/Master/GetStatus
Sent by clients after completing the handshake with the server to request the current state of the score from the master. (See /Status)