Automatic composition and notation in network music environments

Georg Hajdu

AUTOMATIC COMPOSITION AND NOTATION IN NETWORK MUSIC ENVIRONMENTS

Georg Hajdu
Hochschule für Musik und Theater Hamburg
Harvestehuder Weg 12
20148 Hamburg
Germany
hajdu@musikhochschule-hamburg.de

ABSTRACT

Using real-time notation in network music performance environments adds novel dimensions to man-machine interaction. After a 200-year history of algorithmic composition and a 30-year history of network music performance, a number of performance environments have recently been developed which allow performers to read music composed in real-time off a computer monitor. In the pieces written for these environments, the musicians are supposed to either improvise to abstract graphical symbols and/or to sight-read the score in standard music notation. Quintet.net—a network performance environment conceived in 1999 and used for several project involving Internet as well as local network concerts—has built-in notation capabilities, which makes the environment ideal for this type of music. The search for an ideal notation format, for which several known formats were compared, was an important aspect during the development of the Conductor component of Quintet.net—a component that reads and streams parts to the Client and Listener components. In real-time composition, these parts need to be generated automatically. Therefore, different scenarios can be envisaged, which are either automatic or interactive with the players shaping the outcome of a piece by their performance.

1. INTRODUCTION

In this paper, I will outline the principles for automatic and interactive composition and its notation in network music performance environments such as Quintet.net. For this, I will first give a brief account of the history of automatic composition as well as network music performance. This will lead to an overview of current practices and a closer look at Quintet.net with its built-in notation capabilities as well as different aspects of music notation. Three scenarios for automatic and interactive composition in the framework of this environment will be presented, before I conclude my paper with a brief outlook on future developments.

2. HISTORY

Automatic composition has a long history that predates the use of computers by at least two centuries. The use of dice for the composition of short pieces such as the "Würfelwalzer" (ascribed to W.A. Mozart) used to be a popular occupation among music enthusiasts of the 18th century. This interest receded in the romantic and early modern eras just to resurface with renewed momentum in the age of the digital computer. The first example of a piece composed by a computer is the song "Push Berta." Using computers to compose music was an approach that quickly found its way from the vernacular into more serious music thanks to the work of Lejaren Hiller (Ames 1987: 169-185). In Europe, Iannis Xenakis and G.M. Koenig, and later Clarence Barlow, stood out as the main representatives of automatic composition (Barlow 1981). Their works had in common that the required a lengthy and cumbersome transcription process to manually turn the computer printouts into music notation, i.e. the score. The availability of sophisticated notation programs made it possible to also automatize the last steps from algorithmic composition to publishable music scores. With pieces such as "Las Melisas" (1991) and "Servicio a Domicilio" (1991), Roberto Morales (Morales 1992) was among the first composers to implement a system (Escamol), which among other things allowed him to export his compositions to Leland Smith's Score notation program, a task that has become startlingly simple since the advent of OpenMusic (aka Patchwork in the mid-1990s).

Network music performance has a 30+-year history with pioneering work by the "American League of Automatic Music Composers." In a typical setup, the members of the network would use computers programmed individually to react in well-defined ways to the inputs from their fellow members. Hereby, the performance didn't rely on symbolic notation but much rather on the preprogrammed reaction modes of the composers and the intuitive reaction modes of their users (Gresham-Lancaster 1998: 39-44). Since the late 1990s, two Austrian composers Karlheinz Essl and Gerhard Winkler as well as American composer Nick Didkovsky started to employ real-time music notation in their compositions. In 1994, Winkler (Winkler 2004) started development of a Max-based environment, the Realtime-Score, to present a mixture of different graphical elements ranging from standard music notation to animated GUI objects to the performers reading the score off of individual computer monitors. The computers are connected via MIDI to exchange control information. Winkler, who expects the same precision by the performers executing his scores as with

printed scores and parts, first applied his concept to a composition named "KOMA" (1996).

In his piece "Champ d'Action" from 1998, Karlheinz Essl (Essl 1998; Essl and Günther 1998) developed a similar concept of real-time notation, which is also based on Max. He forgoes the use of standard music symbols and employs a reduced set of symbols for 8 different musical structures (points, planes, drone, figures, solo, clouds, trills and repetitions) shaped by four global parameters. The players see the notation on their screens and react to it accordingly. The computers are controlled by a main computer and are also networked via MIDI. Triggers that alter the state of the system can be sent by a conductor or by an external source (e.g. the audience listening to an Internet stream of the performance).

Nick Didkovsky is the co-author of JSML, a Java-based derivative of the HMSL music programming language. JMSL contains the JScore Notation Package (Didkovsky and Burk 2004), a programmable music notation editor (which since recently can also be accessed from inside the Max/MSP programming environment). In his piece "Zero Waste" he uses JMSL to "generate a score in common music notation which is sight-read by the performer" (Didkovsky 2004). The performer is first presented with two measures of algorithmically composed music, whose performance is recorded, transcribed and re-presented to the musician. This leads to an ever-changing loop in which the inaccuracies by the sight-reading of the performer and the transcription of the computer are not only taken into account but also build the foundation of this clever concept piece.

3. QUINTET.NET

Quintet.net is an interactive network performance environment consisting of four components (Client, Server, Listener, Conductor) and one add-on (Viewer). Since its inception in 1999, Quintet.net was used in numerous projects ranging from an Internet opera to local network performances (Hajdu 2005). The Hamburg Network Composers' Collective, a permanent laptop ensemble dedicated to performances of pieces, ranging from free improvisations to fixed compositions (Hajdu 2004), was founded in 2003. A considerable number of pieces have been either transcribed or composed for Quintet.net; a Composition Development Kit (CDK) was added in 2003 to facilitate the creation of new compositions. The CDK consists of several graphical editors such as the Bank Editor, the Score Editor, Condmaker and the Max timeline. The real-time notation built into the environment allows the performers to view their own playing as well as the parts sent by the Conductor.

Quintet.net was conceived with real-time notation in mind, as notation can be extraordinarily useful in Internet performances when musicians are thousands of miles apart from each other. The notation engine also serves another goal: the realization of microtonal compositions. The eighth-tone resolution allows both the exploration and performance of non-standard tuning such as the Bohlen-Pierce scale.

Notation in Quintet.net

Quintet.net handles two types of notation:

- Real-time notation of note events, referred to as *performance notation*
- Notation of parts that are either pre-composed or algorithmically generated in real time, referred to as *score notation*.

Notation in Quintet.net is performed on 5 grand staves, 700 pixels in length and consisting of 70 slots for note events. Small vertical lines demarcate the seven measures. In performance notation, each measure represents the time span of one second; note events are displayed in space notation with quarter-note heads but without stems, beams and further markings. Three levels of dynamics are represented by colored note heads (red, black and blue).

Score notation is more sophisticated and includes different note heads with stems and beams (up to 32nd note beams) as well as rests for rhythmic notation. Markings can be added as formatted text.

	Essl	Winkler	Didko-vsky	Hajdu
Nota-tion	Gra-phical	Graphi-cal and simple music notation	Advan-ced music nota-tion	Real-time and score notation
Net-work	MIDI	MIDI	N/A	TCP/ UDP
Music	Impro-visation	Sight-reading	Sight-reading	Improvi-sation -> sight-reading
Micro-tonality	no	no	no	yes
Exa-mple	Champ d' Action (1998)	KOMA (1994)	Zero Waste (2002)	Mind-Trip (work in progress)

Table 1. Comparison of several performance environments with real-time notation

Compositions for Quintet.net

To perform a Quintet.net composition, a *.cond* file needs to be loaded into the Conductor component, which automatically brings up a timeline and an ensemble of text files, called a score. The individual parts can be either chosen manually by the conductor (the person controlling the Conductor) or automatically from a timeline.

In a real-time composition, parts are generated automatically. Therefore, in addition to opening the timeline, the Conductor loads an entire Max patch with a separate control panel that allows the conductor to view and control the state of the algorithmic process.

Notation formats

During the development of the Conductor, some research on the usability of a notation format needed to be carried out. The evaluation was based on three criteria: Firstly, the composers should be able to use a commercial music notation program to develop their materials for Quintet.net (an import function hasn't been implemented yet, though). Secondly, the automatic creation of parts was supposed to be as simple as possible with as few intermediate steps (formatting/parsing) as possible, and thirdly, it was supposed to support notation of non-standard tunings. For this, five different formats were compared to each other (also see the appendix for a short example coded in each format):

- OpenMusic
- MusicXML
- Enigma
- ABC
- Quintet.net notation format

OpenMusic

IRCAM's Lisp-based OpenMusic (Assayag et al. 1997) composition environment has several powerful music editors, called factories, for microtonal and polyphonic music notation. They require a hierarchical tree structure to be sent to them. Several object classes exist to facilitate the construction of properly formatted lists, whose attributes are implicitly stated by the hierarchy of parentheses (which, typical for Lisp, makes deciphering the data rather difficult). The format would have required major modifications if it were to be considered for Internet streaming.

MusicXML

Recordare's MusicXML is probably the most promising notation exchange format developed to this date (Good 2001). As an extended markup language (XML) it is ideal for Internet streaming and it possesses a large repertoire of attributes, which are capable of describing most types of musical with high accuracy. The large amount of data typical for XML files require broader bandwidth and processing power for the transcription of data into drawing commands which may be a critical resource in network music performance.

Enigma

Coda's music notation program Finale uses a text-based exchange format named Enigma. Enigma which has developed over the course of 20 years is a convoluted format, difficult to interpret (hence the name), but with the advantage of its event list format being capable of employing up to 16 different accidentals per scale step for microtonal notation as well as its relative closeness to drawing commands.

ABC

ABC was developed with basic notation capabilities for medieval and folk tunes in mind (Walshaw). The notation format is very explicit and easy to read. It lacks serious support for microtones, though, which made it a less likely candidate for Quintet.net.

Quintet.net

For what was required, developing a custom format thus seemed the most viable compromise at the time. Due to a lack of commercial editors, the Score_Editor was created as part of the Quintet.net Composition Development Kit.

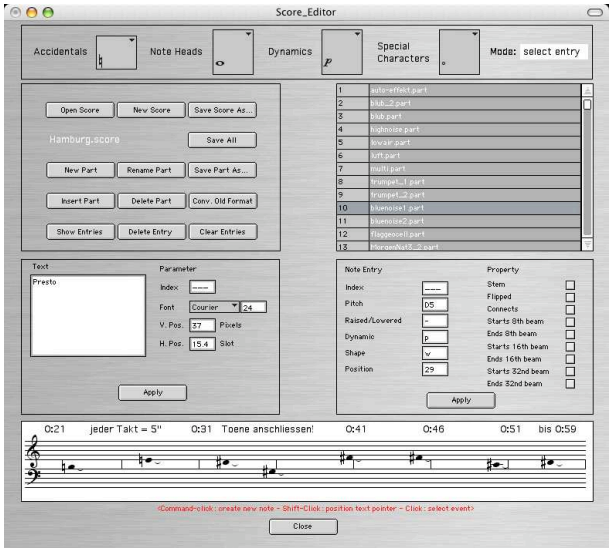


Figure 1. The Score Editor is part of the Composition Development Kit.

	Open-Music	Music-XML	ABC	Enigma	Quin-tet.net
Com-pact code	yes	no	yes	no	yes
8th-note notation	yes	yes	no	yes	yes
Attri-butes	impli-cit	explicit	inter-mediate	implicit	inter-medi-ate
Close to drawing comm-ands	inter-medi-ate	no	inter-mediate	yes	yes

Table 2. Comparison of music notation formats considered for the streaming of Quintet.net parts.

4. SCENARIOS

In the next chapter, I will describe three different scenarios for which I want to employ the terms *scheduled* vs. *interactive real-time* composition.

Scheduled real-time composition

In the last part of his life, John Cage wrote dozens of number pieces which are characterized by an absence of a global score and by individual parts in space notation. Musical events are to be played within so-called "time brackets." As his composition *Five* (1988) was already successfully adapted for and performed with Quintet.net, it is only a small step towards the creation of a *Five*-style real-time composition, whose outcome are determined by the following parameters:

- Total duration of the piece
- Vertical and horizontal pitch organization
- Dynamic (initial dynamics and cresc./decresc.)
- Articulation (legato or cesura)

The algorithm needs to ensure that the parts are being sent out a few seconds before they are supposed to be played.

In a recent piece for recorder and live electronics in 19-tone equal temperament, I employed Clarence Barlow's indispensability formula (Barlow 1987) to generate the pitch material for small and large loops, which are characterized by a hierarchical, self-similar organization. The formula was originally conceived to metric hierarchies for complex meters with arbitrary numbers of strata consisting of prime divisors. A Ligeti-esque machine aesthetic à la *Continuum* characterizes the resulting textures.
In a Quintet.net version for several instruments the following parameters would to be manipulated in real-time:

- Total duration and overall form
- Tempo
- Type of pattern
- Tuning and pitch filter
- Counterpoint
- Rhythmic structure (incl. rests)
- Dynamics
- Articulation

Interactive real-time composition

In this third scenario, the five players influence the outcome of the composition interactively.

a. The five performers are prompted to improvise over an initial pitch material
b. The performance is recorded and played back by a random *walk with aging*. Of the five voice only one is chosen in random order
c. This voice is processed by a texture generator implementing David Huron's principles of *texture space* (Huron 2001: 1 - 64) and the result projected onto the individual computer screens.
d. The performers are prompted to play the music on screen, and the process starts over again from b. until the end of the piece is reached.

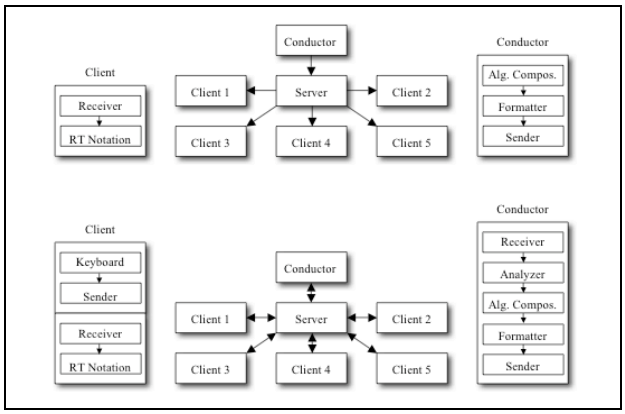


Figure 2: Real-time composition in Quintet.net can either be automatic (above) or interactive (below).

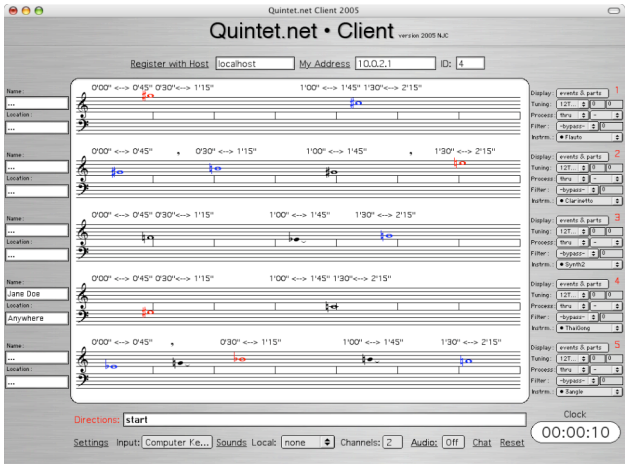


Figure 3. In the "events & parts" display mode, the parts for John Cage's piece *Five* can be viewed at once.

5. OUTLOOK

Automatic and interactive real-time composition in network music environments open up novel ways for the exploration of man-machine interaction. The reliance on music notation and subsequent sight-reading/improvisation adds another dimension to interactive systems, which usually emphasize a more direct form of non-symbolic interaction. The system described in this paper can be used both for performance with electronic instruments in a wide-area network (WAN) or with acoustic instruments in local networks.

6. REFERENCES

Ames, C. 1987. Automated Composition in Retrospect. Leonardo, 20(2): p. 169-185.

Assayag, G., Agon, C., Fineberg J., Hanappe P. 1997. An Object Oriented Visual Environment For Musical Composition, Proceedings of the ICMC 97.

Barlow, C. 1981. Bus Journey to Parametron. Feedback Papers 21-23.

Barlow, C. 1987. Two essays on theory. Computer Music Journal 11: pp. 44-60.

Didkovsky, N. 2004. Recent compositions and performance instrument realized in the Java Music Specification Language. Proceedings from the 2004 International Computer Music Conference.

Didkovsky, N. and Burk, L.B. 2004. Java Music Specification Language, an introduction and overview. Proceedings from the 2004 International Computer Music Conference.

Essl, K. and Günther, B. 1998. Realtime composition. Musik diesseits der Schrift. Positionen. Essl, K. 1998/2000. Champ d'Action. http://www.essl.at/

Good, M. 2001. MusicXML: An Internet-Friendly Format for Sheet Music. 2001 XML Conference Proceedings.

Gresham-Lancaster, S. 1998. The aesthetics and history of the Hub: The effects of changing technology on network computer music. Leonardo Music Journal 8: pp. 39-44.

Hajdu, G. 2004. Composition and Improvisation on the Net. Proceedings SMC'04 Conference, Paris.

Hajdu, G. 2005. Quintet.net: An environment for composing and performing music on the Internet. Leonardo Journal 38(1).

Huron, D. 2001. Tone and Voice: A Derivation of the Rules of Voice-leading from Perceptual Principles. Music Perception 19 (1): pp. 1-64.

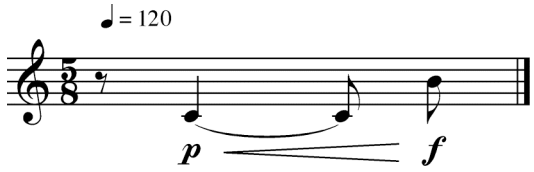
Morales, R. 1992. Non Deterministic Automatons for Composition, Proceedings International Computer Music Conference: pp. 400-401

Walshaw, C. abc standard V1.6. http://staffweb.cms. gre.ac.uk/~c.walshaw/abc2mtex/abc.txt.

Winkler, G. 2004. The realtime-score. A missing link in computer-music performance. SMC'04 Conference Proceedings.

7. APPENDIX

A one-measure example was chosen and the underlying code compared for different music notation formats:



OpenMusic

Various list have to be sent to the inputs of the VOICE factory:

- Rhythm Tree: (5/8 (((5 8) ((5 (-1 2 1.0 1))))))
- Chords: (6000 7100)
- Tempo: 120
- Dynamics and other markings need to be entered manually in the VOICE editor.

Enigma (only note entry pool):

^entries

^eE(1) 0 2 512 0 $81000800 128 0
^eE(2) 1 3 1024 -5 $C0000801 128 1
0 $C0010000
^eE(3) 2 4 512 0 $C0000801 128 1
0 $A0010000
^eE(4) 3 0 512 0 $C0000800 128 1
96 $80010000

MusicXML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE score-partwise PUBLIC "-
//Recordare//DTD MusicXML 1.0 Partwise//EN"
"http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise>
<part-list>
   <score-part id="P1">
     <part-name>MusicXML Part</part-name>
</score-part>
 </part-list>
 <part id="P1">
  <measure number="1">
   <attributes>
    <divisions>8</divisions>
    <key>
     <fifths>0</fifths>
     <mode>major</mode>
    </key>
    <time>
     <beats>5</beats>
     <beat-type>8</beat-type>
    </time>
    <clef>
     <sign>G</sign>
     <line>2</line>
    </clef>
   </attributes>
   <sound tempo="120"/>
   <note>
```

```
      <rest/>
      <duration>4</duration>
      <voice>1</voice>
      <type>eighth</type>
    </note>
    <sound dynamics="54"/>
    <note relative-x="-2">
      <pitch>
        <step>C</step>
        <octave>4</octave>
      </pitch>
      <duration>8</duration>
      <tie type="start"/>
      <voice>1</voice>
      <type>quarter</type>
      <stem>up</stem>
      <notations>
        <tied type="start"/>
        <dynamics placement="below">
          <p/>
        </dynamics>
      </notations>
    </note>
    <direction placement="below">
      <direction-type>
        <wedge relative-y="-7" spread="0"
            type="crescendo"/>
      </direction-type>
      <offset>-6</offset>
    </direction>
    <note>
      <pitch>
        <step>C</step>
        <octave>4</octave>
      </pitch>
      <duration>4</duration>
      <tie type="stop"/>
      <voice>1</voice>
      <type>eighth</type>
      <stem>up</stem>
      <notations>
        <tied type="stop"/>
      </notations>
    </note>
    <direction>
      <direction-type>
        <wedge spread="15" type="stop"/>
      </direction-type>
      <offset>-1</offset>
    </direction>
    <sound dynamics="98"/>
    <note>
      <pitch>
        <step>B</step>
        <octave>4</octave>
      </pitch>
      <duration>4</duration>
      <voice>1</voice>
      <type>eighth</type>
      <stem>down</stem>
        <notations>
          <dynamics placement="below">
            <f/>
          </dynamics>
        </notations>
      </note>
      <barline location="right">
        <bar-style>light-heavy</bar-style>
      </barline>
    </measure>
  </part>
</score-partwise>
```

ABC

```
X:1
T:Example
M:5/8
K:C
"Quarter = 120" z2 C4- C2 B2 |]
w: p cresc. f
```

Quintet.net

```
1, text Times 14 8 36 0;
2, text Times 14 5 25 0;
3, A4 0 p er ^000000000 3.;
4, text Times 14 cresc. 61 4.8;
5, C4 0 p q- ^101000000 5.;
6, C4 0 p q ^101110000 7.;
7, B4 0 f q ^101000000 10.;
```