© positionen, 29/1996, S. 15-19

Karlheinz Essl

Real Time Composition

Komponieren mit Computerunterstützung

Es existieren im wesentlichen zwei grundsätzliche Methoden, den Computer als Werkzeug im Kompositionsprozeß einzusetzen. Die am häufigsten praktizierte verwendet den Rechner als komfortable Schreibmaschine, mit der musikalische Einfälle aufgezeichnet und manipuliert werden können. Als Ergebnis liefern sie entweder eine elektronisch erstellte Partitur, die dann von Musikern gespielt wird (Notationsprogramm), oder eine Folge von Steueranweisungen, mit der elektronische Musikinstrumente angesteuert werden (Sequenzer).

Die andere Methode, der ich mich ausschließlich widme, sieht den Computer als experimentelle Werkbank, um kompositorische Ideen zu entwickeln und zu erproben. Mit Hilfe einer geeigneten Programmiersprache lassen sich kompositorische Algorithmen formulieren und umsetzen. In der Interaktion mit dem Computer kann der Komponist die Tragweite seiner kompositorischen Absichten erkennen, ohne jedoch das Zepter aus der Hand zu geben. Ist er mit den Resultaten nicht einverstanden, kann er das Programm solange modifizieren, bis Intention und Ergebnis zur Deckung gebracht sind. Dabei umschließt das künstlerische Subjekt als erste und letzte Instanz wie eine Klammer den gesamten Kompositionsprozeß: es entwirft ein Modell, analysiert dessen Output und gewinnt dadurch neue Erkenntnisse, die dazu führen mögen, die Ansätze neu zu formulieren oder das Modell zu modifizieren. So kann der Computer zu einer – selbstgeschaffenen - Kontrollinstanz werden, zum unbestechlichen Spiegel der eigenen Vorstellung.

Als ich 1992 ans IRCAM kam, um dort an einem Kompositionsauftrag[1] für Ensemble und die »IRCAM Musical Workstation« (ISPW)[2] zu arbeiten, lernte ich auch die Programmiersprache MAX kennen, welche die graphische Benutzeroberfläche der ISPW darstellt[3].

Die Entwicklung von MAX wurde 1986 am IRCAM begonnen, als man das Paradigma eines Analogstudios, dessen einzelne Komponenten durch Kabelverbindungen zu komplexeren »Patches« verschaltet werden können, auf eine graphische Programmierumgebung übertrug. Die grundlegende Absicht war es, ein Echtzeitsystem zu entwickeln, das vor allem im Live-Einsatz Verwendung finden sollte. Aus einem kleinen Vorrat von System-»Primitives« (graphischen Objekten mit Ein- und Ausgängen) lassen sich durch Verbindungen mit sogenannten »patch chords« höherstrukturierte Objekte erzeugen. Ein solcher »Patch«, dessen interne Komplexität hinter einer »object box« verborgen werden kann, kann nun als eigenes Objekt angesprochen werden. Dadurch lassen sich komplizierte Strukturen aus einigen hochspezialisierten Objekten (Unterprogrammen vergleichbar) erzeugen, die wiederum aus untergeordneten Objekten bestehen usw. Solche Objekte, die Erweiterungen des Sprachschatzes darstellen, können nun in verschiedenen Kontexten verwendet werden, ohne daß man sich jeweils die genaue Kenntnis ihrer internen Organisation vor Augen führen muß. Sie fungieren als »black boxes«, deren Verhalten man zwar genau kennen soll, ohne aber über die programmtechnischen Details Bescheid wissen zu müssen.

Real Time Composition Library (RTC-lib)

MAX dient in erster Linie zur Manipulation von MIDI[4]-Daten, also Steuerinformationen für elektronische Musikinstrumente. Was bislang aber fehlte, war eine Bibliothek mit speziellen Funktionen für »Computer Aided Composition«. Diese Lücke füllt nun die seit 1992 entwickelte »Real Time Composition Library for MAX«. Mit dieser Bibliothek von Softwaremodulen lassen sich Kompositionsalgorithmen auf einer hohen Abstraktionsebene formulieren, ohne daß der Benutzer sich mit untergeordneter Systemprogrammierung abplagen muß. Im Inneren einer solchen Funktion verbirgt sich eine aus MAX-Primitives konstruierte Programmstruktur, die – einmal implementiert – in den verschiedensten MAX-Patches verwendet werden kann (»re-usability«), ohne daß man sich jedesmal den Berechnungsalgorithmus vergegenwärtigen muß. Ist die Funktion einmal ausgetestet und ihr Verhalten bekannt, kann sie wie eine Systemfunktion benutzt werden.

1 *Entsagung* für Flöte, Baßklarinette, präpariertes Klavier, Schlagzeug und 4-kanaliges interaktives Klang-Environment (1991-93) (TONOS Musikverlag Darmstadt).

2 Sie basiert auf einem NeXT-Computer mit spezieller DSP-Hardware und erlaubt die Synthese und Manipulation von Klängen in Echtzeit.

3 Eine objekt-orientierte Entwicklungsumgebung für Echtzeitapplikationen. Die Software existiert in einer mit Klangverarbeitungs-Funktionen angereicherten Variante, die als Benutzeroberfläche dient und in einer zweiten Fassung, die für die Generierung und Manipulation von MIDI-Daten verwandt wird.

4 **M**usical **I**nstruments **D**igital **I**nterface – ein 1983 von verschiedenen Synthesizerherstellern entwickeltes standartisiertes Datenformat zur Steuerung elektronischer Musikinstrumente.

5 Vgl. Karlheinz Essl, *Computer Aided Composition*, in: *Distel* Nr. 46/47 Mensch Maschine, Bozen 1991.

Das Prinzip, durch Programmierung von Softwaremodulen den Grundwortschatz einer Computersprache zu erweitern, habe ich bereits in meinem auf xLOGO basierendem »COMPOSE-Environment« (1988 ff.) angewandt. Dieses System verwende ich in erster Linie zur Generierung von »score listings«, deren symbolisch-numerische Erscheinung zunächst in musikalische Notation übersetzt werden muß, um zu klanglichen Ergebnissen zu gelangen.[5] In MAX hingegen bietet sich die Möglichkeit, alle Operationen in Echtzeit durchzuführen. Damit läßt sich nun endlich ein lang gehegter Traum verwirklichen: die Schaffung einer experimentellen Werkbank zur Entwicklung und Erprobung kompositorischer Strategien, deren Ergebnisse sofort hörbar gemacht werden können. Diese Unmittelbarkeit bringt naturgemäß enorme Vorteile für den Entwicklungsprozeß einer Komposition, da augenblicklich sinnliche Erfahrungen gesammelt werden können. Dadurch läßt sich eine zunächst nur grob skizzierte kompositorische Idee schrittweise verfeinern, bis Intention und Resultat – in einem Rückkopplungsprozeß zwischen Komponist und Computer – zur Deckung gebracht werden (»rapid prototyping«). Der unmittelbare Respons eröffnet aber auch die Möglichkeit, Musikautomaten zu erfinden, die Musik in einem immerwährenden Prozeß in Echtzeit generieren. Diesen Ansatz nannte ich einmal scherzhaft »Realtime Composition«, wohl um die Paradoxie dieses Wortpaares wissend. Mittlerweile hat sich dieser Begriff aber eingebürgert und eine ganze Bibliothek von Softwaremodulen – eben die »Real Time Composition Library« (RTC-lib) – trägt diesen Namen.

Die »Real Time Composition Library« stellt ein *work in progress* dar und erscheint mittlerweile in der Version 2.3. Die Arbeit daran wurde 1992 begonnen, als ich gemeinsam mit Gerhard Eckel (IRCAM) Überlegungen anstellte, welche grundlegenden MAX-Funktionen für musikalische Komposition notwendig wären. Ihm verdanke ich auch die Entwicklung einiger toolbox-Objekte und die Einführung in einen strukturierten und objekt-orientierten Programmierstil. Die Version 1.0 wurde ursprünglich vom IRCAM im Rahmen der »IRCAM Usergroup« vertrieben. Die mittlerweile als Version 2.3 erscheinende RTC-lib (verbessert und stark erweitert, mit Hypertext-artiger Online-Hilfe, Tutorial und Anwendungsbeispielen) ist nunmehr Public Domain-Software und kann im Internet von verschieden ftp-servern bezogen werden.

Die RTC-lib ist in ständiger Veränderung begriffen und reflektiert den momentanen Stand meiner kompositionstheoretischen Auseinandersetzung. Ihre Charakteristika seien hier in Kürze zusammengefaßt:

- ein offenes und erweiterbares, objekt-orientiertes »Realtime Programming Environment« anstelle eines starren, geschlossenen Computerprogrammes,
- modular aufgebaut,
- die RTC-Objekte können in verschiedensten Kontexten verwendet werden (»re-usability of code«),
- Komplexität kann durch Verkapselung (encapsulation) verborgen werden.[6]

Die »Real Time Composition Library« besteht aus 150 Objekten, die sich in zwei grundsätzliche Kategorien unterteilen lassen: *Basic*-Objekte für die Bewältigung programmtechnischer Probleme und *Compose*-Objekte für spezielle kompositorische Fragestellungen. Diese Grundkategorien lassen sich nun weiter klassifizieren.

BASIC
Toolbox: grundlegende low-level Funktionen
Chance: Zufallsoperationen
Lists: Listenoperationen

COMPOSE
Harmony: tonhöhenbezogene Funktionen
Rhythm: rhythmus- und zeitbezogene Funktionen
Envelopes: Hüllkurven und zeitvariante Funktionen (»ramps«)

Die Library erlaubt es dem Anwender, sich mehr auf die kompositorischen Fragestellungen zu konzentrieren, da die Entwicklung von Programmstrukturen auf einem sehr hohen Level erfolgt. Dafür werden dem Benutzer eine Vielzahl von Werkzeugen zur Verfügung gestellt, beispielsweise Zufallsoperationen oder Rhythmus- und Harmoniegeneratoren.

Mittels dieser Werkzeuge lassen sich Strukturgeneratoren konstruieren. Die im Folgenden wiedergegebene Graphik zeigt das Meta-Modell eines Strukturgenerators zur Generierung von Klaviermusik. Es besteht aus vier Patches, (mit einem »p« gekennzeichnete Objektkästchen), die die konkreten Algorithmen einer musikalischen Struktur beinhalten und nicht einer Softwarebibliothek entnommen sind (vgl. Abb. 1).

6 Neben Gerhard Eckel haben auch andere Beiträge zur RTC-lib geleistet, von denen vor allem Peter Elsea (University of California, Santa Barbara), Orm Finnendahl (Berlin), James McCartney (University of Texas), Davis Zicarelli (IRCAM/ Paris), Gary Lee Nelson (Oberlin College) und Charles Baker gedankt sei.

7 Andreas Okopenko, *Lexikon-Roman einer sentimentalen Reise zum Exporteurtreffen in Druden*, Frankfurt am Main, Berlin, Wien 1983.

8 Ebd., S. 6.

9 Diese Methode wurde von Gerhard Eckel gewählt, als er im November 1995 die Lexikon-Sonate als Klanginstallation im »Banff Centre of Arts« (Kanada) präsentierte. Näheres dazu im World-Wide Web unter: http: //www. ping.at /users /essl /bibliogr / lexson-eckel.html

10 So geschehen bei der öffentlichen »Uraufführung« am 10. Februar 1994 im Großen Sendesaal des Österreichischen Rundfunks Wien. Näheres dazu im World-Wide Web unter http://www.ping.at /users /essl /bibliogr / lexson-kunstradio. html.

11 Karlheinz Essl, *Kompositorische Konsequenzen des radikalen Konstruktivismus*, in: *Positionen. Beiträge zur neuen Musik*, Nr. 11, Berlin 1992.

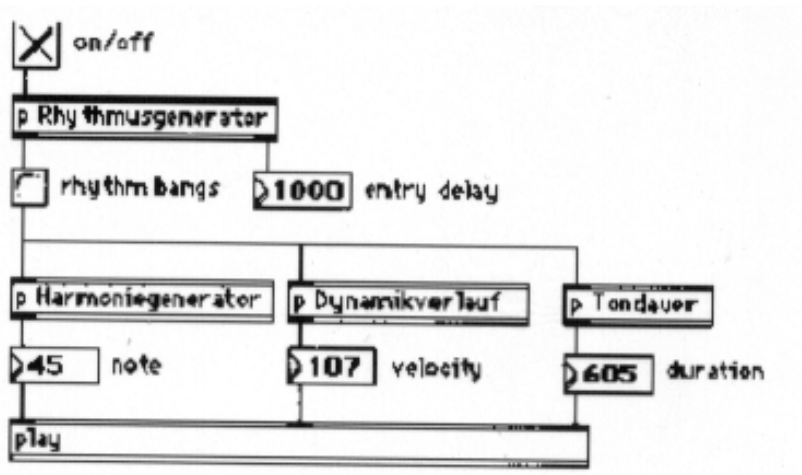

Abb 1: Flußdiagramm eines Meta-Modells für Strukturgeneratoren zur Generierung vomn Klaviermusik

Ein Rhythmusgenerator erzeugt einen bestimmten Rhythmus, indem er rhythmische Impulse (»rhythm bangs«) aussendet. Diese werden an drei weitere Generatoren für Tonhöhe, Tonstärke und Tondauer geschickt und veranlassen diese Objekte, aufgrund ihres implementierten Algorithmus bei jedem Impuls einen entsprechenden Wert zu berechnen. Tonhöhe (»note«), Tonstärke (»velocity«) und Tondauer (»duration«) werden nun vom Objekt »play« zusammengefaßt und als MIDI-Informationen an ein angeschlossenes MIDI-Instrument gesendet und dort gespielt.

Wichtig erscheint mir die Feststellung, daß es sich bei dem obigen Beispiel um ein Meta-Modell handelt, das – bei Verwendung verschiedener Rhythmus-, Harmonie-, Hüllkurven- und Dauerngeneratoren – die Konstruktion völlig unterschiedlicher Strukturgeneratoren erlaubt. Die prinzipielle Organisation des Meta-Modells bleibt davon unangetastet: Der primäre Impulsgeber ist immer der Rhythmus, der mit Tonhöhe, Dynamik und Dauer versehen wird und dadurch erst als musikalische Größe (»Ton«) in Erscheinung tritt. Dies sind nun auch genau die Parameter, die das Klavierspiel beschreiben: nämlich welche Taste (»note«) zu welchem Zeitpunkt (»entry point«) wie stark angeschlagen (»velocity«) und wie lang gehalten wird (»Dauer«).

Das klingt für's erste recht nüchtern und gibt Anlaß zur Befürchtung, daß damit nur sehr hölzerne und mechanisch klingende musikalische Abläufe zu gestalten sind. Daß dem nicht so sein muß, soll anhand der *Lexikon-Sonate* demonstriert werden.

Lexikon-Sonate

Dies ist – so im »Untertitel« – *eine unendliche und interaktive Realtime-Komposition für computergesteuertes Klavier* (1992 ff.). In dem mit Hilfe der RTC-lib komponierten Werk habe ich versucht, die Klaviermusik seit Johann Sebastian Bach (über Beethoven, Liszt, Brahms, Schönberg, Webern, Stockhausen und Boulez) mit ihren Topoi, Sujets und Klischees geistig zu erfassen und dieses Wissen als Strukturgeneratoren zu implementieren. Da das Werk nicht als Notentext existiert, sondern einzig und allein von einem Computerprogramm in Echtzeit generiert wird, habe ich auch Interpretationsparameter wie Rubato, Espressivo, Phrasierung etc. berücksichtigt.

Die Entstehung des Werkes verdankt sich einem Roman, genauer: dem *Lexikon-Roman* (1970)[7] von Andreas Okopenko. Dieses Buch, das die *sentimentale Reise zum Exporteurtreffen in Druden* (so der Untertitel) zum Inhalt hat, kann nicht wie ein herkömmliches Druckwerk von vorne bis hinten gelesen werden. Seine vielen hundert Kapitelchen sind wie in einem Lexikon alphabetisch sortiert. Mittels Verweispfeilen kann der Leser nun wiederum *seine* Reise durch das Buch antreten, und sich nach eigenem Gutdünken durch den Text bewegen. Was heute Dank Internet und Web-Browser schon Allgemeingut geworden ist, war damals noch völlig unbekannt. Ja, nicht einmal der Terminus dafür – HyperText – existierte, als Okopenko sein visionäres Werk verfaßte (dieser wurde einige Jahre später von Ted Nelson »erfunden«).

Die Buchform ist für einen Hypertext freilich das denkbar ungeeignetste Medium. Nur gab es 1970, als das Buch erschien, kaum Computer, geschweige denn World-Wide Web oder Hypertext-Autorensysteme. Deshalb hat sich unter der Federführung von Franz Nahrada ein Gruppe von Computerfachleuten und Medienkünstlern zusammengefunden, die unter dem Namen *Libraries of the Mind* die elektronische Umsetzung des *Lexikon-Romans* (als CD-ROM) in Angriff nahm. Okopenko, der selbst dieser Gruppe angehört, hat nun angeregt, seinen Text multimedial zu erweitern, also Bilder, Photos, Klänge und Musik darin zu integrieren.

So gelangte ich 1992 zu den *Libraries* und wurde mit der Aufgabe betraut, den Musikpart zu gestalten. Nach der ersten Lektüre des Buches wurde mir allerdings rasch klar, daß die ursprünglich vorgesehene Aufgabe – kurze Musikclips für die einzelnen Kapitelchen zu komponieren – nicht zielführend ist. Die Struktur des Buches selbst forderte (so schien es mir) einen gänzlich anderen Weg. Seine potentielle Unendlichkeit und Freiheit der »Wegfindung« – seine explizite Prozessualität – stehen im krassen Widerspruch zur Komposition kleiner, abgeschlossener Einheiten. Außerdem wurde mir bald klar, daß man ein Kapitel auf die verschiedensten Arten lesen kann (andächtig meditierend, flüchtig überschlagend, vor- und zurückblätternd etc.), daß – mit einem Wort – die Verweildauer des Lesers ein unkalkulierbarer Parameter blieb. Ich wollte nun eine Musik schaffen (nicht »schreiben«), die sozusagen das Leserverhalten des »Benutzers« reflektiert. Wie lange auch immer die Verweildauer in einem Kapitel sein mag: während dieser Zeit ertönt eine charakteristische Musik, die sich jedoch verändert, wenn man zu einem anderen Abschnitt wechselt. Dieser Wechsel sollte aber nicht abrupt erfolgen, sondern Teilaspekte des Vorangegangenen weiterführen. Wie in einem Lexikon, wo der Verweis auf ein Stichwort immer noch einen semantischen Rückbezug bedeutet. Okopenko hat dieses »lexikalische Prinzip« im Vorwort des *Lexikon-Romans* wunderbar illustriert: »Wer hat nicht schon im Lexikon, GOLDSCHMINKE nachschlagen wollend, erst einmal den Artikel über GOLDONI, dann den über GOLDREGEN gelesen, dort auf LABURNUM verwiesen, die Einrichtung von LABORATORIEN gestreift, Interesse an der Herstellung eines Chlorkalziumröhrchens gefaßt, das Glasblasen erlernt, dabei einen Wangenriß erlitten, pflasterbeklebt einem Clown geähnelt, nachgedacht, was zum Clown noch fehlt, dabei Blanc und Rouge aufgefunden und so den Gedanken zurückgewonnen, daß er ja GOLDSCHMINKE nachschlagen wollte – was er nun endgültig tat.«[8]

So begann ich nun peu à peu, Topoi aus der Klavierliteratur zu analysieren und sie als Strukturgeneratoren (Module genannt) zu implementieren: Espressivo-Melodien, Akkordstrukturen, Triller, aber auch idiomatische Typen wie Arpeggi und Glissandi. Jedes dieser Module legt ein charakteristisches musikalisches Verhalten an den Tag und generiert unendlich viele Varianten seines implementierten Strukturmodells.

Zur Zeit besteht die *Lexikon-Sonate* aus 24 verschiedenen Modulen. Da sie jedoch als *work in progress* konzipiert ist, kommen laufend neue dazu (vgl. Abb. 2).

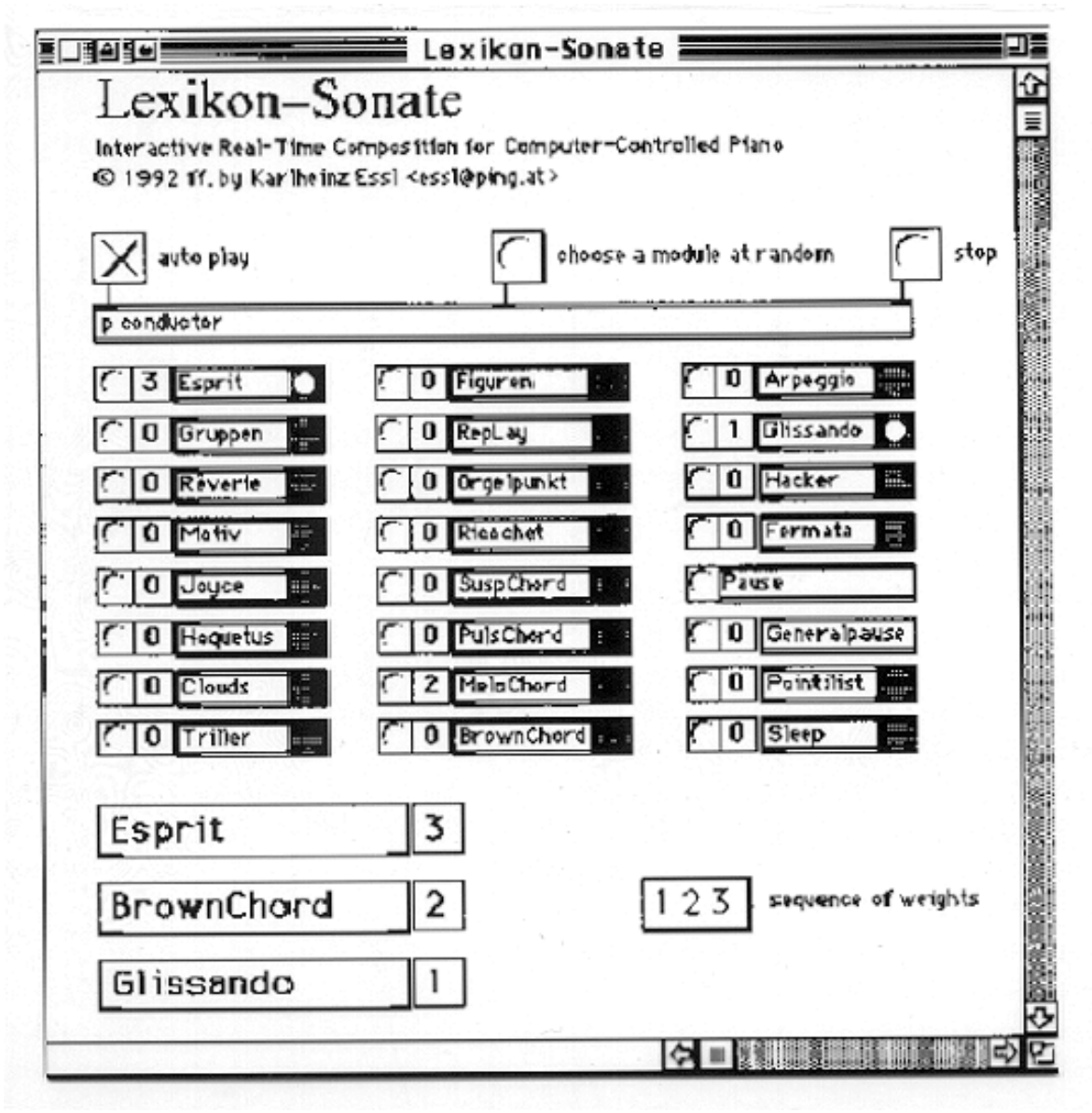

Abb. 2: Benutzeroberfläche der *Lexikon-Sonate*

Wir sehen in der Mitte die 24 verschiedenen Strukturgeneratoren, darüber einen sogenannter »conductor«, der die Auswahl der Module besorgt, und darunter eine dreistufige »Registerbank«, in der nun die selektierten Module von oben nach unten eingetragen werden. Jedem Kästchen der »Registerbank« ist eine Zahl zwischen 1 und 3 zugeordnet – der »weight factor«. In dem obigen Beispiel werden gerade die Module ESPRIT, BROWNCHORD und GLISSANDO miteinander kombiniert, wobei ESPRIT im Vordergrund (3) spielt, BROWNCHORD im Mittelgrund (2) und GLISSANDO im Hintergrund (1). Wenn nun ein neues Modul (z.B. TRILLER) vom »conductor« in die »Registerbank« geschickt wird, rutscht ESPRIT eine Stufe nach unten und erhält den »weight factor« 2. BROWNCHORD wiederum nimmt im untersten Kästchen Platz und bekommt dort den »weight factor« 1 zugewiesen, währenddessen GLISSANDO verschwindet.

Wir sehen also, daß bei Aufruf eines neuen Moduls das »älteste« verschwindet, zwei aber erhalten bleiben, und mit dem neu dazugekommenen kombiniert werden. Dadurch wird ein gleitender formaler Übergang erzielt. Der »conductor« selbst kann auf zwei Arten gesteuert werden: entweder durch einen Auto-Piloten, der ihn in unregelmäßigen Abständen anweist, eine Aktion zu setzen[9], oder aber durch den Benutzer, der willentlich entscheiden kann, wann der »conductor« ein neues Modul auswählen soll[10]. Im ersten Fall ist der sogenannte »toggle« mit der Aufschrift »auto play« einzuschalten, im letzteren wird immer dann, wenn eine Änderung gewünscht wird, der Button »choose a module at random« angeklickt.

Anstelle des »conductors« kann aber auch der Benutzer die Entscheidung treffen, welche Module in die »Registerbank« geschickt werden sollen. Er muß dazu nur auf den Button neben dem gewünschten Strukturgenerator klicken. Darüber hinaus besteht auch noch die Möglichkeit, unter Umgehung der »Registerbank« direkt bestimmte Module auszuwählen und ihnen beliebige »weight factors« zuzuweisen.

Der Mechanismus einer »Registerbank« erlaubt die Kombination von drei verschiedenen Modulen, wobei jedes einen bestimmten Grad von Präsenz (ausgedrückt durch den »weight factor«) aufweist. Die selektierten Strukturgeneratoren laufen synchron als eigenständige Schichten ab, die jedoch nicht miteinander koordiniert sind. Sie durchdringen sich, verschmelzen miteinander oder stoßen sich ab. Es ist schließlich der Hörer, der im Hören eine Synthese im Sinne des »Radikalen Konstruktivismus«[11] leistet und aufgrund seiner individuellen Voraussetzungen den »ästhetischen Gegenstand« in sich Realität werden läßt.

Ein wichtiger Aspekt der *Lexikon-Sonate* besteht in ihrem Vermögen, Allusionen an bereits bestehende Musik hervorzurufen. Obwohl hier keinerlei Zitate verwendet werden (sondern nur algorithmische Beschreibungen von Klaviertopoi), kommt es immer wieder zu déjà-vu-Erlebnissen. Diese tragen zu einer Semantisierung des Gehörten bei und fordern den Hörer zu einer persönlichen Deutung und Sinngebung heraus. In dieser Weise kann das Hören zu einem aktiven Gestaltungsvorgang werden, indem der Hörer das Gehörte in sich zu Musik werden läßt – und damit zum Mitschöpfer wird.

Anhang

Die *Lexikon-Sonate* und die »Real Time Composition Library« (momentan in der Version 2.3 verfügbar) sind Public Domain Software und werden mit MAX 3.0 (IRCAM /OPCODE) vertrieben. Darüber hinaus können sie von folgenden ftp-servern bezogen werden:

Software

ftp://ftp.ircam.fr/pub/IRCAM/programs/max-patches68k/compositions/RTC-lib_2.3.sea.bin

ftp://ftp.samt.co.at/pub/samt/RTC/RTC-lib.sea.bin

Eine Runtime-Version der Lexikon-Sonate (die ohne MAX läuft) kann vom ftp-server des Studios for Advanced Music & Media Technology des Bruckner-Konservatoriums Linz ftp://ftp.samt.co.at/pub/samt/RTC/Lexikon-Sonate_1.5.sea.bin heruntergeladen werden.

Audio

Ein Ausschnitt von der Uraufführung der *Lexikon-Sonate* am 10.2.1994 (mit dem Bösendorfer SE Grand Piano) findet sich auf der CD *Karlheinz Essl: Rudiments* (1995), die von der TONOS Musikverlags GmbH (Ahastr. 9, D-64285 Darmstadt) bezogen werden kann. Eine weitere Live-Aufnahme (diesmal mit einem Yamaha Disklavier) ist auf der von Bruno Degazio herausgegebenen SoundPrint-CD *Roads to Chaos* (Toronto 1996) erhältlich.

NB: Die hier zitierten Texte sowie weitere Informationen zur *Lexikon-Sonate* und zur »Real Time Composition Library« können im World-Wide Web unter der URL http://www.ping.at/users/essl/index.html abgerufen werden.

(Der Beitrag ist eine stark gekürzte Fassung der beiden letzten Kapitel von Karlheinz Essls Schrift *Strukturgeneratoren. Algorithmische Komposition in Echtzeit* (= *Beiträge zur elektronischen Musik* 5), hrsg. von Robert Höldrich und Andreas Weixler, Graz 1996, die beim Institut für Elektronische Musik (IEM) an der Hochschule für Musik und darstellende Kunst in Graz, Jakominstr. 3-5, A-8010 Graz bezogen werden kann.)