2013

Non-Standard Sound Synthesis with Dynamic Models

Valsamakis, Nikolas

Non-Standard Sound Synthesis with Dynamic Models

by

Nikolas Valsamakis

A thesis submitted to the University of Plymouth
In partial fulfillment for the degree of

DOCTOR OF PHILOSOPHY

School of Humanities and Performing Arts
Faculty of Arts

June 2013

Non-Standard Sound Synthesis with Dynamic Models

Nikolas Valsamakis

Abstract

This Thesis proposes three main objectives: (i) to provide the concept of a new generalized non-standard synthesis model that would provide the framework for incorporating other non-standard synthesis approaches; (ii) to explore dynamic sound modeling through the application of new non-standard synthesis techniques and procedures; and (iii) to experiment with dynamic sound synthesis for the creation of novel sound objects.

In order to achieve these objectives, this Thesis introduces a new paradigm for non-standard synthesis that is based in the algorithmic assemblage of minute wave segments to form sound waveforms. This paradigm is called Extended Waveform Segment Synthesis (EWSS) and incorporates a hierarchy of algorithmic models for the generation of microsound structures.

The concepts of EWSS are illustrated with the development and presentation of a novel non-standard synthesis system, the Dynamic Waveform Segment Synthesis (DWSS). DWSS features and combines a variety of algorithmic models for direct synthesis generation: list generation and permutation, tendency masks, trigonometric functions, stochastic functions, chaotic functions and grammars. The core mechanism of DWSS is based in an extended application of Cellular Automata.

The potential of the synthetic capabilities of DWSS is explored in a series of Case Studies where a number of sound object were generated revealing (i) the capabilities of the system to generate sound morphologies belonging to other non-standard synthesis approaches and, (ii) the capabilities of the system of generating novel sound objects with dynamic morphologies.

The introduction of EWSS and DWSS is preceded by an extensive and critical overview on the concepts of microsound synthesis, algorithmic composition, the two cultures of computer music, the heretical approach in composition, non-standard synthesis and sonic emergence along with the thorough examination of algorithmic models and their application in sound synthesis and electroacoustic composition.

This Thesis also proposes (i) a new definition for “algorithmic composition”, (ii) the term “totalistic algorithmic composition”, and (iii) four discrete aspects of non-standard synthesis.

Contents Overview

this page is intentionally left blank

Table of Contents

List of Illustrations

List of Tables

Acknowledgements

I would like to express my gratitude to all the persons that took a significant part in my life during the long road for the completion of this Thesis and especially to:

My supervisor *Eduardo Miranda* for continuous guidance, advice and, encouragement.

The colleagues I met in the Interdisciplinary Centre for Computer Music in Plymouth and especially *Joao* and *Eduardo* for their friendship and hospitality.

My colleagues in the Department of Music Technology & Acoustics in Rethymno and especially *Chrisoula* and *Katerina*.

All my friends in Athens and especially *Dimitris, Kostas M.*, *Kostas K.*, *Yiannis and, Haris*.

All the friends that I met in Rethymno and now are spread around Greece, especially *Nikos, Sofia, Elena, Katerina, Yiannis, and Vaggelis.*

All the friends in Rethymno and especially *Lina, Manolis* H*, Tolis*, Ourania, Marianna, *Aggelos, Alexandros D., Alexandros G., Manolis A., Stella and Vaso.*

My parents Kostas and *Kornelia* for their love and whole-hearted support.

Marina, for her love, encouragement and, support to start it.

Evita, for her love, encouragement and, support to finish it.

Author's Declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award without prior agreement of the Graduate Committee.

A record of activities can be found in Appendix I.

Word count: 48.394

12th of June 2013

……..

Nikolas Valsamakis

Introduction

I was always fascinated by *strange sounds*. I remember me as a kid, when I was visiting during the summer the Greek island of Syros, I was enchanted listening during the night the sounds of nature and I was thrilled by the minute details of the sound of insects. To the same degree I was attracted by the sound of toys, their mechanical sound, when toys unwind, when toys does not work properly.

Through this practice of attentive focusing on the details of sound, I was unintentionally training myself to develop what Pierre Schaeffer call *reduced listening*. Moreover, with this fascination to the sounds of the soundscape and the sounds of machines, I was somehow preparing myself, to encounter years later, the sound world of *electroacoustic music*.

My first encounter with electroacoustic music was during the high school when accidentally listened at the radio to the music of karlheinz Stochausen. This music, completely new for me at the time, struck me immediately. Afterwards, I discovered that the music I was listening was the electroacoustic composition Hymnen (1969). From that moment I wanted to know how to create myself a respective condition through sound composition that would repeat this *primal experience* again.

I was lucky that, some years later at the end of the 80s, I went to the Center of Contemporary Music Research (KSYME) in Athens. There, I was formally introduced to contemporary music composition theory and practice as well as electroacoustic music studio techniques. There, I started learning *music*

programming and constructed my first simple algorithms for automated music generation. It was at KSYME that arrived for the first time in Greece the NeXT computer music system, equipped with the music programming environments of Csound and Cmix.

Although I was already accustomed to the idea of algorithmic composition, it was the music of Iannis Xenakis that provided me with the concept of *formalized music*. Especially, I was fascinated by the computer-generated parts of his landmark electracoustic composition *La Legende d'Eer* (1979) as well as by *GENDY* (1991). Eventually, I decided to dig deeper into this subject and went to the City University in London where I pursued my Master's degree. The subject was precisely "Aesthetics and Techniques in the Electroacoustic Music of Iannis Xenakis". There I acquired an insight to the concept of *direct waveform synthesis* through algorithmic procedures in a computer and the idea of Dynamic Stochastic Synthesis that is behind the GENDY computer music system.

Later, I encountered the radical concepts and the *non-standard computer music* techniques of composers like Herbert Brun, Gottfried Michael Koenig, Paul Berg, Agostino di Scipio and others. In parallel, I was investigating algorithms derived from *chaos theory* and *evolutionary artificial systems* and started to develop my own *microsound* synthesis systems and experiment with novel sound sonorities in electroacoustic music composition.

The idea of creating directly the sound waveform by the application of custom defined algorithmic procedures opened to me the question on *a novel formalization of sound modeling based on minute waveform segments*. A

research on such a formalization would not only encompass ideas and techniques behind Xenakis's GENDY system along with the work of Brun, Koening and others but would also provide an *original paradigm* of sound modeling that investigate concepts of evolutionary computing and chaos theory.

This research aspires to *provide the answer to a number of questions* that emerge from the involvement with concepts and practices on the above fields:

- is it possible to provide a new generalized sound synthesis model that is based on waveform segments and that it would incorporate the basic concepts of non-standard segment-synthesis techniques proposed by Xenakis, Brun, Koening, and others?
- Is it possible to use the concept of Cellular Automata as a generalized framework that extends the concept of stochastic waveform evolution proposed by Xenakis in the GENDY system?
- Is it possible to use various algorithmic models (for example oscillating functions, chaotic systems, stochastic systems) along with more "traditional" Cellular Automata rules and achieve musically interesting results?
- Is it possible to use higher-order algorithms (for example oscillating functions, chaotic systems, stochastic systems, L-systems) to automatically control multiple processes by Cellular Automata in order to generate even more complex sequences of waveform segments in a musical interesting perspective?

This research answer the above questions by proposing the model of *Extended*

Waveform Segment Synthesis, by providing the paradigm of *Dynamic Waveform Segment Synthesis,* and by exploring the construction of a variety of synthesized sounds within a number of *case studies*.

The nature and the goals of this research project are double-sided. One side is "*scientific*": the proposal of a new sound synthesis model. The other side is "*artistic*": to provide a "compositional instrument" that is good in creating original computer music. For my point of view these two perspectives are the two sides of the same coin. If we are interested in creating original music we must develop new concepts and tools and by doing so we contribute to broaden our knowledge.

Research objectives

Formally, this Thesis proposes three main objectives:

I. to provide the concept of a new generalized non-standard synthesis model based on minute waveform segments that would provide the framework for incorporating other non-standard synthesis approaches;

II. to explore sound modeling through the application of Cellular Automata and other dynamic algorithms as the engine of a new dynamic non-standard synthesis technique for microsound composition;

III. to experiment with dynamic sound synthesis for the creation of novel sound objects.

As part of the process of currying out the above objectives, this Thesis will extensively cope with and critically examine the concepts of microsound, algorithmic composition, and non-standard synthesis along with many others.

These concepts will be encapsulated in a new theoretical paradigm for non-standard synthesis that is based in the algorithmic assemblage of minute wave segments to form sound waveforms. This paradigm is called Extended Waveform Segment Synthesis (EWSS) and intergrades both abstract sound structures along with structures derived from recorded sound material.

A consequent aspiration of this Thesis is to realize the basic concepts of the extended waveform segment synthesis paradigm and to implement them in a sound synthesis application. This application, which eventually is called Dynamic Waveform Segment Synthesis (DWSS), would utilize a combination of dynamic algorithmic generative and transformative models for the generation, evolution and assemblage of sound segments. One aim is to propose the application of continuous Cellular Automata as the algorithmic basis for the evolutionary transformation of microsound structures.

Finally, we intend to experiment and demonstrate, in a number of case studies, how DWSS is capable of generating sound objects that features basic characteristics of other non-standard synthesis approaches and most importantly, how it can generate new sound objects featuring morphologies that belong to the heretical currents of contemporary computer music creativity: *Dynamic Sound Synthesis*

Chapter overview

This Thesis is divided into 14 Chapters

“Chapter 1: Microsound” introduces to the notion of the hierarchy of time scale levels in relation to composition, and focuses to the musical and technical

concept of microsound which form the actual sonical time scale level of the main subjects of this Thesis.

“Chapter 2: Algorithmic Composition and Computers” discusses in depth the concept of the application of rules and procedures in music composition along with the utilization of computer technology.

“Chapter 3: Historical Foundations” presents information theory along with the early algorithmic music approaches of Hiller, Xenakis, and Koening as the historical foundation of algorithmic composition and non-standard synthesis.

“Chapter 4: The two cultures of computer music” examine the notions of disguised / explicit computer music and example based / rule based composition as the basis for the differentiation between standard and non-standard synthesis.

“Chapter 5: Non-standard synthesis: foundations” discusses in depth the concept of non-standard synthesis and provides historical definitions of the term along with an extensive collection of quotations.

“Chapter 6: Non-standard synthesis: Historical examples” presents the implementation of non-standard synthesis paradigms of Xenakis, Koening, Brun, Berg, Holtzman and others.

“Chapter 7: Sonic Emergence” examines the notion of emergence in relation to the hierarchical levels of sound synthesis.

“Chapter 8: Computers, cognition and music analysis” discusses basic concepts and methodologies of the scientific field of cognitive musicology in relation to

apprehension and the analysis of sound objects generated by non-standard synthesis.

“Chapter 9: Algorithmic models” presents the basic algorithmic models that are utilized in sound synthesis and music composition along with historical examples.

“Chapter 10: Extended Waveform Segment Synthesis (EWSS)” introduces to the paradigm of Extended Waveform Segment Synthesis and proposes it as a new generalized approach on non-standard synthesis with waveform segments.

“Chapter 11: Dynamic Waveform Segment Synthesis (DWSS)” introduces to the first implementation of the dynamical non-standard sound synthesis concepts of EWSS.

“Chapter 12: EWSS & DWSS in the contexts of non-standard synthesis” describes the novelty of the concepts of EWSS and the potentiality in synthesizing original sound objects with DWSS and place them side-by-side with other non-standard synthesis approaches.

“Chapter 13: DWSS: case studies” presents a number of case studies that reveal some of the synthetic capabilities of DWSS.

“Chapter 14: Conclusions” highlights the contributions to knowledge introduced in the Thesis and make recommendations for future advancements in the field.

this page is intentionally left blank

1 Microsound

With the advent of technology, composers are able to apply music design principles beyond the level of note or sound object representations. Computer technology, along with the digital representation of sound, enabled the precise control of sound formation. Concepts and formalizations of computer aided algorithmic composition can be potentially applied from the construction of whole pieces, sections and phrases down to the structuring of the sound objects themselves. In this regard, the notion of microsound enables the integration of the various composition levels. Phil Thomson suggests:

> *"microsound generally works with an integration of time scales, relating the sub-note level with the level of sound gestures, sections, movements and whole pieces. As such, it so far seems to be the approach to electroacoustic music and sound design that comes closest to realizing the long- standing dream of 'total composition': composition of everything from the overall form to the individual sounds themselves."* [Thomson 2004, pp. 1].

Curtis Roads, a principal theorist and composer of the field, devoted a complete book with the same title, *Microsound,* to present various concepts and processes for microsonic design and transformation. Roads states in the introduction of the book [Roads 2001, pp.vii]:

> *"Beneath the level of the note lies the realm of microsound, of sound particles… Microsonic techniques dissolve the rigid bricks of music architecture - the notes - into a more fluid and supple medium. Sounds may coalesce, evaporate, or mutate into other sounds. The sensations of point, pulse (regular series of points), line (tone), and surface (texture) appear as the density of particles increases. Sparse emissions leave rhythmic traces. When the particles line up in rapid succession, they*

induce the illusion of tone continuity that we call pitch. As the particles meander, they flow into streams and rivulets. Dense agglomerations of particles form swirling sound clouds whose shapes evolve over time."

By the term "microsound" we refer both to the scale of music levels that lies below the sound object (microsound levels, e.g. micro-level, sample-level etc) as well as to the generative and transformational procedures that construct the structural elements of the sound object (microsound synthesis).

1.1 *The hierarchy of the time-level scale in relation to composition*

Curtis Roads suggested a *hierarchy of time scale levels* in relation to music composition. These levels, starting from the longest, are as follows [Roads 2001]: infinite, supra, macro, meso, sound object, micro, sample, subsample, and infinitesimal. Four of them (macro, meso, sound object, micro, sample) are of particular interest since they correspond to structural hierarchies of single computer music compositions:

- *macro level*: the overall structure of a composition or macroform.
- *meso level*: phrases or sequences of sound objects.
- *sound object*: the notion of the basic unit of music structuring. It is a generalization of the traditional concept of musical notes to include complex sound events.
- *Micro level*: audio transient phenomena and particle structural elements of sound objects.
- *Sample level*: Elementary digital audio representations. A single sample impulse is the shortest possible sound duration in the computer environment.

The rest of the hierarchical levels suggested by C.Roads either extend below any possible sound digital representation (subsample and infinitesimal) or above the time span of a single composition (supra, infinite).

In this Thesis we are focusing in the construction of sound objects by non-standard means. Our non-standard synthesis proposals operate in the micro-level and should be considered belonging to the arsenal of microsound composition. Therefore we are especially interested in three sound levels: the sound object, the micro-level, and the sample-level.

1.1.1 The sound object

The notion of *sound object* includes any possible sound as basic music structuring unit. The sound object extends the rather abstract notion of musical note. The term of the sound object used in this Thesis is rather broad and should not be confused with the term "*objet sonore*" proposed by Pierre Schaeffer and in which the sound source is supposed not to get recognized by the listener [Kane 2007].

Sound objects may have different properties and thus their perception may be varying. Their difference is usually comprehended through their not common properties. Properties may change over time making the sound object dynamic. In this aspect sound objects are considered *heterogeneous*. The heterogeneous concept of the sound object is contrasted to the homogeneity of the note, which can be described by four common abstract properties of: timbre (referring to a particular instrument), pitch, dynamic, and duration. Curtis Roads suggests that the "loss" of homogeneity is offset by opening up electronic music into a variety of dynamic sound formations [Roads 2001, pp.336]:

> *"To adopt the universe of heterogeneous sound objects is to be cast into a strange new land without conventional language. The terrain of this land is non-homogeneous, pocked by fractured disjunctions (intermittencies) and nonlinear transitions from one perceived state to another."*

1.1.1.1 Spectromorphology and reduced listening of sound objects

Pierre Schaeffer, in his classic book *Traite des object musicaux* (1966), suggested a taxonomy of sound objects according to their acoustic *morphology*. Denis Smalley coined the term "*spectromorphology*" to describe perceived morphological developments in sound object spectra over time [Smalley 1986, 1997]. Many composers consider the morphological study of sound objects seminal to the theory of electroacoustic music. Dennis Smalley comments on spectromorphology:

> *"I have developed the concepts and terminology of spectromorphology as tools for describing and analysing listening experience. The two parts of the term refer to the interaction between sound spectra (spectro-) and the ways they change and are shaped through time (-morphology). The spectro- cannot exist without the -morphology and vice versa: some- thing has to be shaped, and a shape must have sonic content… A spectromorphological approach sets out spectral and morphological models and processes, and provides a framework for understanding structural relations and behaviours as experienced in the temporal flux of the music."* [Smalley 1997, pp.107]

The morphological studies of sound objects make use of a specialized listening mode that focuses on the detail and the quality of the sound itself independently of its cause and of its meaning. This mode of listening is termed by Pierre Schaeffer "*reduced listening*". Michel Chion includes reduced listening, along with casual and semantic, in his three modes of listening [Chion 1994]. Dennis

Smalley characterizes reduced listening as: “an abstract, relatively objective process, a microscopic, intrinsic listening” [Smalley 1997]. As we have already seen, music perception is conditioned by cultural and historical conventions and interpretations. As Michel Chion suggests, “perception is not a purely individual phenomenon, since it partakes of a particular kind of objectivity; that of shared perceptions. And it is in this objectivity-born-of-intersubjectivity that reduced listening, as Schaeffer defined it, should be situated.” [Chion 1994].

The morphological studies of sound objects belong to the phenomenological school. We have already seen that the phenomenological school of thought is principally focused in the experience. Michel Chion explains reduced listening as a phenomenological approach that targets to the materiality of the sound object stripped off any semantic aspect:

> *“listening intention targets the event which the sound object is itself (and not to which it refers) and the values which it carries in itself (and not the ones it suggests). In "ordinary" listening the sound is always treated as a vehicle. Reduced listening is therefore an "anti-natural" process, which goes against all conditioning. The act of removing all our habitual references in listening is a voluntary and artificial act, which allows us to clarify many phenomena implicit in our perception. Thus, the name reduced listening refers to the notion of phenomenological reduction (Époché), because it consists to some extent of stripping the perception of sound of everything that is not "it itself" in order to hear only the sound, in its materiality, its substance, its perceivable dimensions.”* [Chion 1983, pp.31]

However, reduced listening usually requires listening to the specific sound more than one time. Computer music technology allows repeated listening that eventually detaches the spectromorphological aspects of sound from any reference to its cause and its meaning. Computer generated sound objects that

are abstract modeled, as is the case of non-standard synthesis, are distanced from any semantic reference already from their mode of production. Moreover, any reference to the source, directly address to the computer algorithmic process that generated the sound object.

1.1.2 The micro-level

Microsound synthesis is of special interest for this Thesis: non-standard synthesis operates in its entirety by applying unconventional algorithmic procedures for sonic composition in the micro-level. The microlevel “embraces transient audio phenomena, a broad class of sounds that extends from the threshold of timbre perception (several hundred microseconds) up to the duration of short sound objects (~100 msec). It spans the boundary between the audio frequency range… and the infrasonic frequency range…” [Roads 2001]. Micro-level transient phenomena are widely encountered throughout the natural soundscape, its geophony and its biophony [Truax 2001], as well as to the detailed gestures on the playing of any musical instruments. Curtis Roads presents twenty-one different terms used in the scientific literature of acoustics and signal processing concerning micro-level concepts [Roads 2001].

The major theoretical basis of micro-level formations lies in the granular concept of sound, first proposed by physicist Denis Gabor. In the granular concept, the basic micro-level sound unit is the grain. A sound object is represented as a concatenation of grains. The granular concept of sound combines into a single representation the time-varying waveform and the static frequency spectrum.

The Gabor theorem as well as the granular synthesis model of sound is discussed in more detail in section (12.2.1.2).

1.1.2.1 Synthesis and transformation on the micro-level

Iannis Xenakis suggests on the possibilities of granular synthesis and transformation [Xenakis 1992, pp.47]:

> *"In fact within human limits, using all sorts of manipulations with these grain clusters, we can hope to produce not only the sounds of classical instruments and elastic bodies, and those sounds generally preferred in concrete music, but also sonic perturbations with evolutions, unparalleled and unimaginable until now. The basis of the timbre structures and transformations will have nothing in common with what has been known until now."*

Xenakis soon became aware of the necessity for a global organisation principle of grains. As granular synthesis of sound uses thousands of grains, the composer's focus must shift from the attributes of individual grains to the attributes of global grain formations. Xenakis himself proposed macroscopic grain arrangement by means of statistical and set-theory operations:

> *"…to work like architects on the sonic material in order to construct complex sounds and evolutions of these entities means that we must use macroscopic methods of analysis and construction. Microsounds and elementary grains have no importance on the scale which we have chosen. Only groups of grains and the characteristics of these groups have any meaning."* [Xenakis 1992, pp.49-50]

Curtis Roads carried a thorough research in various microsound synthesis concepts, techniques and implementations. Concerning the global organization of grain units, Roads proposed the following possibilities [Roads 2001]:

- Matrices and Screens
- Pitch-Synchronous Granular Synthesis
- Synchronous Granular Synthesis
- Quasi-Synchronous Granular Synthesis
- Asynchronous Granular Synthesis
- Physical Models
- Algorithmic Models
- Streams and Clouds of Granulated Samples

Beyond granular synthesis, Curtis Roads proposed a variety of microsound techniques or “varieties of particle synthesis” [Roads 2001]:

- Glisson Synthesis
- Grainlet Synthesis
- Trainlet Synthesis
- Pulsar Synthesis
- Graphic and Sonographic Synthesis of Microsound
- Particle-Based Formant Synthesis (FOF, Vosim, Window)
- Synthesis by Transient Drawing
- Particle Cloning Synthesis
- Physical Models of Particles
- Abstract Models of Particles

Synthesis techniques and transformation techniques on the micro-level are closely related. They interchange concepts and procedures. Their main difference is that usually synthesis begins with micro-level scale material (single waveforms, impulses, noise bursts) while transformation begins with sound object scale material. Curtis Roads proposed a number of microsound transformation techniques [Roads 2001]:

- Micromontage (by graphical sound editing, script, and algorithmic process)
- Granulation

- Pitch Shifting
- Time Stretching
- Filtering
- Dynamics Processing
- Waveset and Wavecycle Distortions
- Convolution of Microsounds
- Spatialization of Sound Particles
- Sonographic Transformations

From the above microsound synthesis and transformation possibilities, our research focuses and investigates Abstract Models of Particles in general and non-standard synthesis in particular.

1.1.3 The sample-level

The sample-level is the level of direct signal representation into discrete information of impulse units. Each impulse unit represents one instance within a sequence of samples that digitally represent a sound. The timing of impulses is determined by the sampling rate of the digital clock of the system. Each individual impulse unit carries so little information that no sense of timbre can be derived. Timbral significance emerges only when a relatively large number of impulses are ordered into a sequence.

Operations on the sample-level are governed by the theoretical and mathematical basis of the sampling theorem, proposed by Nyquist and others [Nyquist 1928]. The sampling theorem provides the scientific framework for any digital representation of sound and audio processing by computers. On section (12.1) we will discuss the sampling theorem as the basis of elementary models of sound.

Some non-standard synthesis approaches operate directly on the sample level for the construction of sound objects without necessitating in the formation and organization of intermediate micro-level structures. Herbert Brun, was one of the first composers that proposed the concept of composing the sound instead of composing with sound:

> *"For some time now it has become possible to use a combination of analog and digital computers and converters for the analysis and synthesis of sound. As such a system will store or transmit information at the rate of 40,000 samples per second, even the most complex waveforms in the audio-frequency range can be scanned and registered or be recorded on audio tape. This… allows, at last, the composition of timbre, instead of with timbre. In a sense, one may call it a continuation of much which has been done in the electronic music studio, only on a different scale. The composer has the possibility of extending his compositional control down to elements of sound lasting only 1/20,000 of a second." [Brun 1970, pp.36]*

2 Algorithmic Composition and Computers

Algorithmic approaches have been applied in composition during the music history for centuries. The production of music with automatic instruments, algorithms and procedural rules has a long tradition. In a parallel pathway, algorithmic processes are found in the development of technique and technology. Algorithms are inherent in software implementations of computer systems. With the advent and spread of computer applications in music, algorithms play an increasing role in music representation and production. But can we term any music produced with the help of computers as “computer music”?

Martin Supper defined *computer music* as “music that cannot be created without the use of computers” [Supper 2001]. However, this definition is too vague. Nowadays we have specialized software and hardware for sound recording, sequencing, mixing, restoration or notation along with automated score generation, sound synthesis, sound processing, or interactive performances. Computer technology has an increasing tendency in covering virtually every aspect of music production. There are less and less aspects of music production and performance that are distinct from any computer involvement. In this perspective, virtually any music tends to be applicable under the term “computer music”. Thus, the above definition is so general that tends to become meaningless.

However, there are often diverse and innovative systems that break with established musical paradigms, elaborate compositional resources and propose novel models of compositional design. These systems involve some sort of

conceptual attitude towards sonic creation with the computer. They involve some manifestation of the creative modes and generative processes with the help of the machine. In this perspective, algorithmic methods take inherent part in the creative process.

A number of terms have been used to define usually overlapping and occasionally identical concepts in the field: algorithmic composition, computer music, computer aided composition, computer assisted composition, computer composing, programmed music, automated composition. Various researchers attempted to distinguish these terms [Spiegel 1989, Cope 1991, Burns 1994, Truax 1999, Miranda 2000].

The term *algorithm* does not have a generally accepted definition. There is an ongoing debate between researchers in formalizing the term. In an informal attempt, we could define algorithm as "a set of rules that must be followed when solving a particular problem" [Oxford 2006].

In our effort for a definition, *algorithmic composition consists of musical concepts that are formalized and employed by the composer in rules and procedures that generate elements, parts or the whole musical work*. Algorithmic procedures may be applied at a variety of compositional levels, from the macro-structure down to the micro-sonic detail. Algorithmic composition is an inextricable amalgam of concepts, procedures and human choices. Algorithmic procedures are interpreted in the musical domain and the generated results are then evaluated and assessed by the musical preferences of composers.

Nevertheless, the term algorithmic composition is not specific to the use of the computer. Although the spread of computer technology had a great impact on the development of algorithmic composition, the use of the computer is not a prerequisite for it. Many historical and contemporary compositional approaches are evident for this [Nierhaus 2008]. Therefore, we consider as more appropriate the hybrid term *Computer Aided Algorithmic Composition* proposed by Ariza [2005]. With this term, we overcome the generality of other terms, like “computer aided composition” or “computer music”, that include any manner to facilitate the musical output (eg. sequencing or notation) and are not considerably specific for employing generative algorithms with a computer. Additionally, this term makes clear the distinct link between the procedures (algorithmic) and the implementation framework (computer).

We have already seen that computer aided algorithmic procedures may be applied either to the composition of whole musical structures or to the synthesis of individual sounds. In the above concept, sound generation itself can be considered as compositional activity. Stockhausen writes, “every sound is the result of a compositional act” [Stockhausen, 1963]. Di Scipio complements, “synthesis can often be thought of as micro-level composition.” [Di Scipio 1995b]. The difference between composition and synthesis is rather a difference in the time level the algorithmic operation takes place than of a kind. This is evident if we take into account that the two terms “com-position” and “syn-thesis” are etymologically synonymous in their respective language of origin (Latin and Greek respectively).

2.1 *Totalistic Algorithmic Composition*

The music practice with computer aided algorithmic composition is an aesthetic current that is followed by a number of composers with diverse approaches. Composers apply algorithmic procedures both in different aspects and in various degrees of composition: from score generation to direct waveform synthesis and from the sketchy computation of individual music parts or layers to the entire automatic generation of composition.

Compositional interest in the formalization of musical procedures as well as the advent of computers and their enormous computability, led some composers to support the idea of computational automation of the entire composition.

The composer York Holler, although never used the term algorithmic, utilizes compositional operations that are capable of generating entire compositions. Holler comments on this approach [Holler 1984, pp.35]:

> *"The work of art seemed to me to be above all an organism, like an organicoenergizing system, comparable to a living organism in nature. In such a system, all elements are linked by functional relations; they do not result from an arbitrary formulation, but from evolution of a process."*

Kristina Burns state further, "algorithmic composition in its strictest sense would involve a program in which the composition is generated entirely by a series of rules that solve a problem based on recursion" [Burns 1992].

We call "*totalistic*" the algorithmic music approach that utilizes computer procedures for the generation of the entire composition up to its finest detail. Other have used similar terms like "integrity" [Laske 1981], "pure" [Ariza 2005] or "rigorous" algorithmic composition [Hoffman 2009].

The output of totalistic algorithmic synthesis is considered as music codified in some symbolic form, either in the form of music notation or in the form of sound waveform data. Later, musicians may perform the notation or an audio system may played back the soundfile. Some may suggest that a strict definition of algorithmic composition includes only the latter case, if the composer wants to avoid performance and human interpretation by musicians. Peter Hoffman suggests, “the entirety of a musical artwork is computationally defined, up to and including every atom of the sound itself” [Hoffman 2009]. However, this approach ignores that there are no “neutral” audio systems and that sound data reproduction also involves some form of “interpretation” by the inherent electro-mechanical characteristics of the playback system. This debate has its roots to the rather philosophical issue whether music exists in its symbolic form or needs to be listened.

Following this radical approach of algorithmic music, Gregory Kramer considers the composition as being the audification of machine instructions and performance [Kramer 1994]. On the other extreme, the group of computer artsist, subscribing with the single pseudonym of Netochka Nezvanova, considers music composition as the algorithmic process of the sonification of digital data [Nezvanova 2000].

Some suggest that totalistic algorithmic music provide some degree of aesthetic, compositional or musical integrity. Otto Laske state accordingly, “if the composer wishes to maintain the algorithmic integrity of the output, a consistent strategy for resolving such conflicts should be developed” [Laske 1981]. Sever Tipei goes even further saying that subsequent interference by the composer to the generated music is “foolish, because it cancels the most

important gain offered by this kind of endeavor, that of a qualitatively approach to composition".

The above views tend to ignore the role of the composer and the issue of human interpretation and decision-making, both during construction and during operation of the algorithmic system.

Horacio Vaggione argues integrity not only in algorithmic composition but also on any musical formalism and reductionism [Vaggione 2001, pp.54]:

> *"Composers, especially those using computers, have learned — sometimes painfully — that the formal rigor of a generative function does not guarantee by itself the musical coherence of a result. Music cannot be confused with (or reduced to) a formalized discipline: even if music actually uses knowledge and tools coming from formalized disciplines, formalization does not play a foundational role in regard to musical processes."*

Curtis Roads supports this view and challenges whether formalisms are necessarily cognitive validated [1996, pp.846]:

> *"Simply, because certain parameters of a piece… conform to an arbitrary set of axioms is no guarantee that the listener will hear consistency or originality in the final product. Musical consistency and originality are cognitive categories for which little theory yet exists."*

Some non-standard synthesis approaches could be categorized as totalistic algorithmic composition. For example, the GENDY synthesis system by Xenakis, is a totalistic algorithmic composition system that is capable of generating all the levels of composition. With GENDY Xenakis was able to compose the macrostructure, individual musical layers as well as to synthesize the sounds in microsonic detail using stochastic processes. GENDY, along with

other non-standard synthesis systems, will be further discussed in section (13.1.1).

3 Historical and Theoretical Foundations

As we have already seen, algorithmic composition models have been proposed and applied long before the advent of computer technology. The first computer machines appeared in early 1940s including the Mark-I and the ENIAC mainframes. Almost a decade later introduced the first research in composition with the aid of a computer.

In parallel with the advent of the first computer technology we have the rise of particular theories and implementations that strongly influenced the historical development of computer-aided algorithmic composition. On the one hand we have the theories on Information and Communication, mainly proposed by Wiener, Shannon, and Weaver. On the other hand we have the early algorithmic compositional experiments by Hiller, Xenakis and Koenig.

3.1 *Information Theory*

Harry Nyquist in 1924 provided preliminary approaches on the concept of information and how it can be transmitted by a communication system. However, it was Claude E. Shannon who provided the foundation of information theory or otherwise communication theory with the publication of the seminal text *A Mathematical Theory of Communication* in 1949 [Shannon 1949]. Shannon was influenced by the theories of cybernetics of Norbert Wiener as well as by the concept of entropy in thermodynamics.

The basic notions in information theory are the transmission of a message from a source to a receiver through a channel. Shannon also developed the concepts of information entropy, redundancy, and introduced as well the term “bit” as a

unit of information. Information theory is closely related with various theoretical and technological advancements like adaptive systems, anticipatory systems, artificial intelligence, complex systems, complexity science, cybernetics, informatics and machine learning. Claude E. Shannon, together with Harry Nyquist, proposed the sampling theorem, which is the basis for sound representation into digital data sequence [Gleick 2011].

Information theory, is concerned with quantitative aspects of information. It does not have anything to do with qualitative aspects like semantics and meaning.

Information theory makes extensive use of probabilities for the generation and formation of messages. Information is formatted by selecting from a finite set of symbols with the help of Markov chains [Shannon 1949].

The concept of information, the application of probabilities as well as the demonstrations of algorithmic text generation were seminal influences for early algorithmic music systems.

3.2 *Early Algorithmic Music Approaches*

The early works of Lejaren Hiller, Iannis Xenakis and Gottfried Michael Koenig constitutes the historical foundation for the development of the field of computer-aided algorithmic composition. Their work was an influence for many generations of composers as well as for the development of various computer music systems. To this influence contributed the detailed documentation they provided on their theories and their implementations.

3.2.1 L.Hiller and the Illiac Suite

Lejaren Hiller together with Leonard Isaacson started experimenting with algorithmic music procedures as early as 1955. Isaacson composed the *Illiac Suite for String Quartet* (1955-56) with the help of a computer program he wrote together with Hiller. They documented their investigations in the book *Experimental Music* [Hiller *et al* 1959]. *Illiac Suite* is organized in four-movements, each demonstrating a different algorithmic approach. Although the music application of the computer was a novelty along with the use of stochastic methods, the musical inscription was constraint in historical formalisms of style imitation and pedagogy. In Experiment One they created 500 monophonic melodies. In Experiment Two they employed the 18th century Fux model for classical counterpoint to select between randomly generated diatonic pitches. Their composition method called “try-again process” and was described as “the extraction of order out of a chaotic environment” [Hiller 1956]. For the end of the second part, they applied the emulation of “simple modulations and a movement towards a final cadence” [Hiller *et al* 1958]. Experiment Three was generated to imitate constraint chromatic music. In that movement they used Markov chains for the selection of adjacent intervals and harmonies along with Heinrich Schenker’s analysis methods on the hierarchical structure [Hiller 1956, 1970, Hiller *et al* 1958, 1959].

3.2.2 Xenakis and the Stochastic Music Program

The style imitative approach of Hiller and Isaacson was highly criticized by Xenakis, as a “failure to confront the real musical problems of the present” [Matossian 1986]. Xenakis considered the work of Hiller and Isaacson more as a “musicological research”. Instead, he insisted that his own Stochastic Music

Program was the first computer music program developed for the composition of “absolute” music [Varga 1996].

Xenakis was always looking for music originality. This trend was expressed in his research and experimentations on new principles and procedures of music composition. Xenakis published in 1955 an article called *the crisis of serial music* where he strongly criticized the theory and practice of that dominant at the time musical direction. Xenakis proposed as an alternative the application of stochastic methods in composition [Harley 2004]. Xenakis himself mentions:

> *“I think this has been my main contribution to contemporary music: masses of sounds controlled like clouds by means of probabilities that shape the clouds statistically” [Varga 1997, pp.142].*

Pithoprakta (1955-56) for orchestra was the first composition in which he applied stochastic procedures in composition. Xenakis developed the Stochastic Music Program (SMP) in 1962 as an early attempt to automate the compositional process with a computer. SMP took advantage of statistical methods for the control of various musical parameters over different compositional levels (e.g. length of sections or note parameters). The following aspects of the composition were handled by SMP [Xenakis 1992]:

1. The average duration of each section
2. The density of notes in each section
3. The classification of instruments into timbre classes for each section
4. The timbre of each note
5. The pitch of each note
6. The glissando speed of each note
7. The duration of each note
8. The intensity evolution of each note

Some of the above parameters are interdependent: the pitch class is dependant from the timbre class, pitch is dependant from the timbre and instrument classes, and the glissando is calculated by pitch, speed and duration parameters. The generated musical events had the form of alphanumerical lists and required manual transcription into notation.

Xenakis thought the algorithmic structure of SMP as a generalized formalism of composition that is capable of generating a number of works. With the aid of SMP Xenakis composed a number of important instrumental works including *ST/4* (1956-62), *St/10* (1956-62), *St/48* (1962), *Atrées* (1962), and *Morsima-Amorsima* (1962) as well as some sections of *Eonta* (1963-64).

During that period, Xenakis was already considering the possibility of using stochastic techniques in direct waveform synthesis:

> *"Although this program gives a satisfactory solution to the minimal structure, it is, however, necessary to jump to the stage of pure composition by coupling a digital-to-analogue converter to the computer. The numerical calculations would then be changed into sound, whose internal organization had been conceived beforehand" [Xenakis 1992, pp.144].*

This initial proposal was later developed into Dynamic Waveform Synthesis [Xenakis 1992].

3.2.3 G-M. Koenig and the PR1 & PR2 Programs

G-M. Koening developed the computer program Project1 (PR1) in 1964, which followed by Project2 (PR2) in 1967. In parallel with Xenakis, he wanted to cope with compositional problems that he encountered with serialism. The main objective was the "calculation of musical structures" with the combination of

serial and aleatoric procedures as discrete compositional elements [Koenig 1971a].

In PR1 Koenig applied his concept of the “regular” and “irregular” opposites, that he called the “R1 priciple”. In PR1 the R1 principle is actually a measure for generating between periodic and aperiodic sequences of the following parameters:

1 Instrument or instrument group
2 entry delay that work as metronomic units
3 pitch that defines the intervals of three-tone groups
4 octave register
5 dynamics

On each parameter the composer assigned a list of desired values. The program used two discrete procedures, the “series” (random selection with replacement) and the “alea” (random selection without replacement), for the random selection and permutation of elements from the lists and the generation of parameter sequences.

Similarly to the SMP program, the PR1 program generates a score-table. The composer interprets the score-table and produces a score for any number of instruments.

The PR2 computer program was more general and allowed greater control for the composer. The composer defines a list with eight parameters which are grouped in the following categories [Koenig 1971a]:

- Instrument: lists of instrument names; each melody or percussion instrument is defined in terms of its pitch, duration and dynamic range.

- Rhythm: lists for entry delays, durations, rests and tempi.
- Harmony: a choice of three harmonic principles: chord list, row, interval table.
- Dynamics: list of dynamic indications.
- Articulation: list of articulation modes.

Parameters are hierarchical interdependent: “once a parameter has been composed… the following parameters must adapt to those already composed” [Koenig 1971b]. Hierarchy is either defined by the composer or generated randomly.

In PR2 Koenig applied six discrete list selection and permutation procedures:

- series (random selection without replacement)
- alea (random selection with replacement)
- ratio (weighted random selection)
- group (selection with group of elements repetition)
- tendency (random selection within boundaries)
- sequence (composer’s selection of individual elements)

The program requested input data for more than 60 questions. Then generated the final score-tables autonomously, without any further intervention by the composer.

In both programs, Koenig applied his concept of the fluctuation of parameters between the opposites of periodic and aperiodic for the control of musical repetition and variation. In PR2 Koenig provided more complex control by applying the concept of “stockpiles” of parameters in list selections. Selection and permutation procedures grouped list elements into intermediary “stockpiles”

of various depths. The different list selection methods are capable of generating stockpiles with a variety of sequences and distributions. Further selections from specific stockpiles were possible by generating tightly constraint element selections [1971b].

Koenig composed nine works with the aid of PR1: *Projekt 1 - Version 1* (1965-1966), *Projekt 1 - Version 3* (1967), *Output* (1979), *Segmente 1-7* (1982), Segmente 99-105 (1982), *3 Asko Stü*cke (1982), *Segmente 92-98* (1983), *Segmente 85-91* (1984), and *Beitrag* (1985-1986). Two more works composed with the aid of PR2: *Übung* (1969) and *60 Blätter* (1992).

4 The two cultures of computer music

Peter Hoffman proposed the concept of “two cultures of computer music” to describe diverse and sometimes antagonistic practices and aesthetics in the field of computer composition. He called these two cultures as “disguised” and “explicit” computer music [Hoffman 2009].

The *disguised computer music*, which is the majority trend, is interested in emulating aspects of music making practice within the established cultural framework. This trend tries to “humanize” the machine and produce “natural” and “beautiful” sound, terms that derive from the dominant western culture positivist concepts and aesthetics. One representative of the disguised approach and seminal researcher of physical modeling techniques is Julius O. Smith. In a paper titled “Viewpoints on the history of digital synthesis” Smith speculates about the future of synthesis models and projects his positivist perspective by envisaging the disappearance of non-standard synthesis along with other abstract methods:

> *“The most straightforward way to obtain interesting sounds is to draw on past instrument technology or natural sounds… The best way we know to understand a sonic transformation is to study its effect on the short-time spectrum, where the spectrum-analysis parameters are tuned to match the characteristics of hearing as closely as possible. Thus, it appears inevitable that sampling synthesis will migrate toward spectral modeling. If abstract methods disappear and sampling synthesis is absorbed into spectral modeling, this leaves only two categories: physical-modeling and spectral- modeling. This boils all synthesis techniques down to those which model either the source or the receiver of the sound.” [Smith, 1991, p.9]*

The *explicit computer music* is interested in conceptualizing the use of machines and taking the technical means into account, stressing the

computational aspect by using rigorous formalisms, and experimenting with what is beyond the established culture in a radical approach. Theorist Michael Hamman discusses the explicit compositional practice where the composer designs algorithmic processes as part of his own particular compositional tools with which the compositional process is carried out [Hamman 1999a, pp.7]:

> *"In this context, composition consists in enabling a composer to compose the very means (i.e. elements within the task environment) by which compositional activity might be carried out. The operations and operands which define the task environment themselves become explicit by virtue of their being rendered as formalized elements within a symbolic system such as a computer. Such elements include programs, data structures as well as visual and interactive components. In such an environment, compositional hypothesis are determinative of both the artifacts being designed and the explicitly formulated processes by means of which those artifacts are designed."*

Hoffman's differentiation between implicit and explicit computer music cultures is very close to the distinction Otto Laske made between, example-based composition and rule-based composition [Laske 1991]. In *example-based composition*, the composer relies on existing materials and models, on historical experience and practice, for the design of musical processes that produce the musical result. In *rule-based composition*, the composer defines and implements his own rules and procedures that generate the music material. In the latter approach, composers "focus on the pro-active, rather than the re-active, aspect of their activity, [giving] them a chance to choose, rather than suffer, their process" [Laske 1991]. Hamman suggests that what distinguishes rule-based composition is the taking into foreground and the conscious use of all the formalizing procedures as part of the creative process. This is called *externalization and objectification* of the composition processes: "what

distinguishes rule-based approaches is that there is an effort to represent otherwise internal processes externally; to objectify them so that, as observable objects and processes, they may be consciously moulded and manipulated" [Hamman 2000].

Eventually, the differentiation between implicit and explicit computer music cultures as well as between example-based and rule-based composition lead to the concept of standard and non-standard synthesis of sound. Non-standard synthesis, in its various implementations, is a radical approach of rule-based composition belonging to the explicit computer music culture. The distinction between these two sound synthesis approaches will be discussed in chapter (7).

4.1 *A Heretical Approach to Computer Music*

From the beginning, electroacoustic music provided a heretical or anorthodox paradigm for the re-interpretation and (ab)-use of technology as a means to the creative process. Early electroacoustic techniques involved the application of standard radio equipment in ways that went far beyond their initial conception, design and mode of operation. For example, tape speed as well as tape speed stability (wow and flutter) is considered as important quality factors for the frequency response of the tape machine regarding the faithful capturing and reproduction of sound events. However, composers took control over both tape speed and speed stability as part of their creative means. Pitch change, time flow reversal, time flow repetition (looping), and sound processes like phasing and flanging were but a few of the artistic means that led the unorthodox use of

the tape machine. Agostino Di Scipio comments about the heretical use of technology in early electroacoustic music [Di Scipio 1997, pp.11]:

> *"In the 1940s and 1950s Elektronische Musik and Musique Concrete were born by an unprecedented re-interpretation of technical instruments which were solely meant for scientific measurements and control. In that case, means of reproduction, control and storage were bent to a form of creative production - of poiesis - which was completely alien to their original technical code."*

The same heretic attitude by composers towards technology in their creative process can be found throughout the history of electroacoustic music, either electronic or computer based. Regarding computer music, the heretical use of technology belongs to the explicit computer music culture. Michael Hamman suggests that computers do not only allow for the exploration of novel music ideas but as well as for the critical assessment of the compositional process itself [Hamman 2002, pp.93]:

> *"The computer provided the kinds of tools that allowed the composer to explore more deeply, the very conceptual frames in which musical ideas might be imagined and realized. It enabled the composer to critically examine and assess the musical result, the means by which the result came about, and how the two are conceptually and generatively related"*

Composers invent new approaches that go beyond standard techniques and technologies in order to control their creative means in an original way instead of interpreting them in a culturally established and usually commercial way. G-M. Koenig comments on his non-standard approach in computer music [Koenig in Roads 1985, pp.573]:

> *"Primarily I am very annoyed with composers using the most modern tools of music making, like electric music, voltage*

control, even computers, and making twelve-tone series for instance, or trying to imitate existing instruments. This has, of course, its scientific value, but not necessarily a creative value in new music making... So just to be able to avoid that, to open up new fields of sounds you would not be able to produce or would not think of describing in classical terms, I have chosen this non-standard approach."

5 Non-standard Synthesis: Foundations

If we take the closely related concepts of explicit computer music and rule-based computer music down to the microsonic level, we are eventually reaching the realm of abstract sound modeling and non-standard synthesis.

Abstract modeling in general and non-standard synthesis in particular are detached from any representational approach. Non-standard synthesis does not try to emulate either the acoustical properties of the sound phenomenon, as is the case with acoustic modeling, or the physics underlying the sound production of the source, as is the case with physical modeling (a taxonomy of sound modeling approaches is presented in chapter 10). The generated sounds in non-standard synthesis are artificial abstract algorithmic microsonic structures, without any reference to any natural phenomenon or any pre-existing model. The abstract modeling of non-standard synthesis should be thought as an avant-garde approach to sound construction with direct similarities to abstract painting or abstract film.

The non-standard notion of sound has its foundation in the very concept of sound computability. It changes the descriptive function of the sampling theorem into a productive function. Non-standard synthesis is closer than any other synthesis model to a reductionist concept of sound. In this approach, sound is the audible representation of a creative compositional process and applies aesthetics and formalizations down to the microlevel of sound structuring. Curtis Roads states on non-standard techniques [Roads 1996, pp.319]:

> *"They are nonstandard in the sense that were conceived for the production of new electronic sounds, rather than starting from simulations of traditional instrument tones. They are motivated by compositional aesthetics, by the desire of creative imagination for fresh musical resources."*

Non-standard synthesis approaches are abstract instead of immitative. They are creative instead of reproductive. Their aesthetics are rather radical and usually stretch beyond any culturally established concept of "beauty" towards more absurd sonorities. Their musical practice to sonic composition is exploratory and experimental since they are dissociated from any already given, tried and tested sound theory or model. Agostino Di Scipio describes non-standard synthesis and recognizes the exploratory attitude one has to follow when coping with sound design [Di Scipio 1994b, pp.4]:

> *"Models of standard synthesis are instances of known theories of sound; one utilizes them in his/her own model of musical design. In contrast, non-standard models instantiate a possible theory of sound; one explores them, and learns how they can mediate the sonic structure."*
>
> *"…these embody an arbitrarily devised process and reflect an abstract model corresponding to no generalisable (acoustic) theory. In the computer implementation of such processes, the control-structure features primarily compositional – rather than physical or psychophysical - parameters."*

Holtzman, Koenig, Berg, Brun and Xenakis were the first composers who experimented with non-standard synthesis and developed a heretical approach towards sound technology. The compositional efforts of these composers was not intended in the generation of pleasant, natural or even plausible sounds. Merely, they were mostly interested in creating compositional processes, which would manifest novel sounds and sonic structures. Holtzman and Berg utilized sequences of arithmetic and logic operations on binary numbers. Koenig

applied techniques derived from serial and aleatoric composition for the construction of sound waveform segments. Brun used abstract operations on direct waveform segment synthesis. Finally, Xenakis utilized stochastic processes for the formation of sound wavecycles.

5.1 *Historical definitions of non-standard synthesis*

Between 1978 and 1980, S.R. Holtzman in the Department of Computer Science of the University of Edinburgh as well as G-M. Koenig and Paul Berg in the Institute of Sonology in Utrecht carried seminal research and provided the basic theoretical framework for non-standard synthesis. Initial implementations of non-standard synthesis techniques found their way in the Automated Digital Sound Synthesis Instrument (S-R. Holtzman), SSP (G-M. Koenig), and PILE (P. Berg) computer music programs. An earlier non-standard approach in composition has been carried by G-M. Koenig in his Project 1 (PR-1) and Project 2 (PR-2) computer programs. This group of composers formed the "school" of Utrecht, for whom the specificity of machine computation provided the subject of music composition. Barry Truax states about the compositional approach of Utrecht school [Truax 1999, pp.24]:

> *"In more colloquial parlance, these approaches continued what has been called the 'hard edge Utrecht school' of electronic music, known for its abrasive sound quality and uncompromising compositional structures"*

During the research for this Thesis, we visited the library of the Institute of Sonology, currently located in the Royal Conservatory of The Hague, and carried out a thorough research on research papers and technical reports on the above computer music programs, surveyed the concept of non-standard

synthesis and collected characteristic and important passages concerning the two terms and their applications.

Holtzman himself was the first to propose the terminology of standard and non-standard synthesis techniques. According to his definition, non-standard synthesis is a bottom-up approach that is based on computer instructions. The latter determine direcly defined relationships between sound samples that do not refer to any higher-level acoustic model. The following rather long quote dates from 1979 and is derived from a historical research report with the title *"A description of an automated digital synthesis instrument – D.A.I. Research Report No.59"* written by S-R.Holtzman to the Institute of Sonology. This rather long quote is highly revealing and provides an original conception of non-standard synthesis:

> *"The synthesis technique used in our system rests on 'non-standard' approach to digital synthesis: based on digital processes, synthesis is built around a technique of applying sequences of… instructions to samples… The synthesis possibilities are considered 'idiomatic' to the extend that the technique is limited and build around a particular machine cpu architecture. The meaning of non-standard is twofold: firstly it is meant to suggest the noises this technique tends to generate non-standard sounds in terms of the sound repertoire of contemporary instrumental or electronic music; and secondly that the approach to describing a sound is in terms of basic digital processes rather than the standard descriptions that use acoustic based sound models (e.g. Fourier models, Frequency modulation, etc.) and traditional concepts of frequency, pitch, timbre, overtone structure, etc.*
>
> *In standard models, the relationships between samples are created given some descriptions of a desired timbre, frequency, attack etc… What is considered the essential qualifying feature of standard synthesis techniques is that, working 'top-down', the samples and relationships between them are the function of some higher-level acoustic model.*
>
> *In non-standard synthesis, we refer to a process where the samples are determined not in the basis of some description of*

> *timbre, frequency, etc. but rather, samples are related only one to another, relationships created determining the timbre, frequency etc.; 'relating only one to another' suggests that the relationships are diacritically defined and do not refer to some superordinate model or function… The samples are conceived of in terms of machine-instructions rather than on the basis of some acoustic theory.*
>
> *Standard approaches to digital synthesis are characterized by an implementation process where, given a description of a sound in terms of some acoustic model machine instructions are ordered in such a way so as to simulate the sound described; the non-standard approach, given a set of instructions, relates them one to another in terms of a system which makes no reference to some superordinate model, i.e. is self-contained, and the relationships formed are themselves the description of the sound."* [Holtzman 1978, pp.1-2]

The PILE computer language designed by Paul Berg uses sequences of computer instructions to generate directly the sound waveforms. This algorithmic aproach and aesthetic on sound synthesis is characteristic of composers around the Institute of Sonology. Berg writes about the PILE computer language [Berg 1979, pp.30]:

> *"PILE instructions are based on groups of machine operations, not on a particular acoustic model. Parameters such as frequency, timbre, envelope, and duration are not specifically referenced… A myriad of sound-synthesis programs are based on models related to instrumental music or the design of a traditional analog electronic studio… They all require the use of a computer because of the magnitude of the task... It is a valid, but it is certainly not the most interesting one. More interesting ones are: to hear that which without the computer could not be heard; to think that which without the computer would not be thought; to learn that which without the computer would not be learned… Computers produce and manipulate numbers and other symbolic data very quickly. This could be considered the idiom of the computer and used as the basis for musical work with the computer."*

G-M. Koenig developed the SSP computer music program for non-standard sound synthesis in 1979 at the Institute of Sonology. Koenig comments on

computers, computing, music, the abstracted definition of sound and the exploratory approach in composition:

> *"The computer acts as a sound-generating instrument sui generis, not imitating mechanical instruments or theoretical acoustic models." [Koenig 1980, pp.111]*
>
> *"We should rewrite music theory in binary terms. Create a new grammar for computers. Something which is adapted to the kind of systematic thinking of the computer world. Nothing vague. Either 0 or 1." [Koenig quoted in Holtzman 1994, pp.241]*
>
> *"As opposed to programmes based on stationary spectra or familiar types of sounds, the composer will be able to construct the waveform from amplitude and time-values. The sound will thus be the result of a compositional proc- ess, as is otherwise the structure made up of sounds." [Koenig 1970b, pp.113-114]*

Moreovere, Koenig refers to his non-standard approach in a 1979 interview with Curtis Roads [Roads 1985, pp.572]:

> *"This program uses what we call the "non-standard approach" to sound synthesis. That means not referring to a given acoustic model but rather describing the waveform in terms of amplitude values and time values. My first intention was to go away from the classical instrumental definitions of sound in terms of loudness, pitch, and duration and so on, because then you would refer to musical elements which are not necessarily the elements of the language of today. To explore a new field of sound possibilities I thought it would be best to close the classical descriptions of sound and open up an experimental field in which you would really have to start again. It would be the task of a later time or other people to map the new possibilities to the old experiences."*

Finnally, Barry Truax comments on the computer music programs of Koenig and Berg [Truax 1999, pp.24]:

> *"They are termed non-standard because they are based on no known acoustic principle or parameter, but rather on basic microlevel data manipulations – expressed either as amplitude/time sequences or in the logical form of machine language operations that result in such sequences"*

5.2 *The idiomatic character of non-standard synthesis*

Non-standard systems are highly personalized, computer oriented, explicit approaches of digital sound synthesis. In these sense they are highly *idiomatic*. Stephen Holtzman highlights the rise of new idiomatic aesthetics with the utilization of computers in music composition [Holtzman 1994, pp.240]:

> *"From a creative perspective, what is interesting is not how well computers can emulate traditional human models for performing their tasks and solving problems, but, rather, the new territory that computers will reveal. What are the new possibilities opened by computers? What ideas and means of expression will we discover that are only conceivable with computers? What new models will we develop for viewing the world in light of computers? What means of expression are idiomatic for computers?"*

Non-standard synthesis provides a very idiomatic paradigm of music composition whose very essence is derived from the notion of machine computability itself. Questioning the idiomatic qualities of non-standard synthesis, we propose in this Thesis four discrete aspects: formal, sonical, personal and machine oriented.

- Each non-standard synthesis system proposes a novel approach in *formal* sound modeling. The algorithm is an abstract conception and does not rely on any pre-existing theory or higher order model. Any non-standard formalization is a new proposal for a unique sound model.

- The *sound world* of each non-standard synthesis is highly distinguishable. Non-standard systems are capable of generating new

sound structures of unheard-off sounds. Usually, the non-standard algorithmic models are restricted to a narrow range of sonorities that are highly differentiated and immediate recognizable by any other model.

- Non-standard synthesis techniques are original formalized conceptions that define a very *personal* artistic idiom. Each technique is based on a specialized formalization that is capable of generating a sonic world that eventually characterizes the artistic identity of their designer artist.

- Finally, non-standard synthesis is idiomatic to specific machine or at least that was the case for the early examples. Historically non-standard systems were implemented on particular computer hardware and written for specific CPU. Nowadays algorithmic systems are usually transferable to various computer environments. However, some implementations still remain *machine oriented* (e.g. algorithms designed on the KYMA system).

6 Non-standard Synthesis: Historical Examples

For the last 40 years, on every decade new paradigms for non-standard synthesis appear. Although those paradigms are limited in number, the insistence on a heretical approach on computer music show the continuous interest in non-standard synthesis.

During the mid 1970s, the first wave of non-standard synthesis systems appeared, with Iannis Xenakis's *Dynamic Stochastic Synthesis*, G.M. Koenig's *SSP*, Paul Berg's *PILE*, S.R. Holtzman's *Automated Digital Sound Synthesis Instrument*, and Herbert Brün's *SAWDUST*.

The second wave, appeared during the 1990s, with Iannis Xenakis's *GENDY*, Arun Chandra's *TrikTraks* and *Wigout*, Gordon Monro's *Fractal Interpolation Waveforms* and, Jaques Chareyron's *LASy*.

A third wave is formed during the 2000s around the research of Agistino di Scipio on *Iterated Nonlinear Functions*.

In recent years, the interest for non-standard synthesis is revitalized. This takes place especially around the concept of Iannis Xenakis's Dynamic Stochastic Synthesis. This is seen in the various implementations and variations of the GENDY system as well as with the musicological research carried on with it. Other implementations provide a different approach, for example Stelios Manousakis's *Non-standard Sound Synthesis with L-Systems.*

Our own research, also motivated by the GENDY system but enormously extends it and goes beyond it, asserts its own place in the current of non-standard synthesis.

6.1 *Dynamic Stochastic Synthesis*

Iannis Xenakis presented the concept of Dynamic Stochastic Synthesis in 1972 in the chapter “New Proposals in Microsound Structure” of his book “Formalized Music”. In that chapter Xenakis discussed several conceptual approaches for composing sound in the sample level.

In Dynamic Stochastic Synthesis, waveforms are constructed by linearly interpolating a set of breakpoints. A pair of duration and amplitude values defines each breakpoint. At every repetition of the waveform, the amplitude and duration values are varied stochastically with the application of a pair of random walks (stochastic models, including random walks, are presented particularly in chapter 11.3). Any probability distribution can be employed to determine the size and direction of the steps (e.g., uniform, Gaussian, exponential, Poisson, Cauchy, arc sin, logistic, nested distributions). There are as many pairs of random walks as the number of breakpoints in the waveform. Each random walk is forced to remain within a predefined value range by means of two elastic barriers that reflect excessive values back into the specified range. These barriers provide control over the frequency and amplitude of the waveform. Xenakis envisioned a sound synthesis method ranging between determinism and indeterminism [Xenakis 1985, pp.179-180]:

> *“Starting from a pressure-time curve [...] one may continue by repeating this curve and at the same time injecting stochastic modification into it after every repetition. This stochastic modification is chosen so as to produce the statistically continuous negation of the original period, affecting the timbre, pitch, rhythm, intensity, and general evolution simultaneously. [...] It becomes the job of the composer to master, with intuition and reason at the same time, the doses of [this negation]. In*

other words, one establishes an entire range between two poles — determinism, which corresponds to strict periodicity, and indeterminism, which corresponds to constant renewal."

Xenakis first investigated the possibilities of Stochastic Dynamic Synthesis at the Centre for Mathematical and Automated Music (CMAM) at the Indiana University. Some of the sounds of the composition of La Légende d'Eer (1979) were created with this method.

6.1.1 GENDY

The concept of synthesis through stochastic variation of the sound waveform was further explored with the GENDY program in mid 1990s. GENDY is the acronym for "Génération Dynamique Stochastique" or in English "Dynamic Stochastic Synthesis" Basically, the major difference between the new implementation of the algorithm and the previous one is the use of a pair of second order random walks. The role of the second order random walks was to control the successive step widths of the primary pair random walks: the probability distribution generates the step sizes of the first random walk (the primary random walk); the successive positions of the primary random walk are the step sizes of the secondary random walk [Xenakis 1992, Serra 1993, Hofmann 2000, 2009]

GENDY provided also the automated control of the macrostructure of the composition through the Parag program with the application of another stochastic functions.

Xenakis used the GENDYN program for the composition of Gendy3 (1991) and S.709 (1994).

6.1.2 Recent variations

During the recent years there is a revitalization of the compositional and musicological interest on GENDY. Peter Hoffman is the major researcher on the subject. As part of his research he implemented the New GENDY Program and analyzed the two compositions of Xenakis through resynthesis [Hofmann 2000, 2001, 2004, 2009]. Other researchers and composers recreated the GENDY algorithm or used it as the basis for their own algorithm extensions: Jaeho Chang's *XENAK* [Chang 1999], Alberto de DeCampo's *miniGENDY* [Hoffman 2011], Andrew Brown's *Interactive Dynamic Stochastic Synthesizer* [Brown 2005], Sergio Luque [2009], Luc Döbereiner [2008, 2009a, 2009b], Nick Collins's *iGENDYN* [Hoffman 2011], and Angelo Bello's *GenLab* [Hoffman 2011].

6.2 *SSP*

SSP is a computer sound synthesis program designed by composer G.M. Koenig at the Institute of Sonology in Utrecht in 1972. Previous design plans date back to the 1960s. SSP is based on the concept that "musical sounds may be described as a function of amplitude over time" [Koenig, 1971]. Many of the functions of "Project 1" (PR1) and "Project 2" (PR2) programs have been applied in SSP for the generation of sound itself. Similar to the List–Table-Ensemble principle of PR2, "predetermined precompositional rules are used to build a sound with the basic units of amplitude values and time values. The rules are either aleatoric procedures or the direct enumeration of a sequence of values." [Banks *et al* 1979]. Amplitude and time values were specified, selected and then joined into waveform segments. Segments were represented in core

memory as breakpoints. The 12-bit representation of the system allowed 4096 different amplitude values. Time was expressed in microseconds with a minimum value of 38 microseconds. In SSP, the composer worked interactively with a number of elementary functions like: ALEA, SERIES, RATIO, TENDENCY, SEQUENCE, and GROUP. All these functions make use of a rather small number of parameters for the construction of larger chunks of sequences of waveform segments [Berg 1978, 1979b, Banks *et al* 1979, Berg *et al* 1980]. Among them, an important and distinguished feature was the application of tendency masks for the ordering of segments. According to the compositional experience of Paul Berg with SSP [Berg 2009, pp.84]:

> *The ordering of segments using tendency masks was particularly successful. A wide selection of segments would result in a noisy sound structure. Narrow masks led to unstable sounds within a confined frequency region. Masks moving from narrow to wide could produce dramatic transitions between these two extremes.*

In SSP, the user's task could be described as:

- LIST the source material (amplitude and time values)
- SELECT all or part of the lists
- construct SEGMENTS from the selection
- order the segments (PERMUTATIONS)
- listen to the chosen order
- return to any of the preceding steps for further refinement.

Three pieces were composed with this program: *Mandolin* (1979) by Paul Berg, *Blue Flute* (1979) by Robert Rowe and, *One Room to Another* (1979) by David Theriaul.

6.3 *Instruction Synthesis: PILE*

PILE was a computer language for direct sound synthesis designed by Paul Berg at the Institute of Sonology in Utrecht in 1972. The basis of PILE was a group of machine instructions for the PDP-15 computer. Curtis Roads suggested the term 'Instruction Synthesis' to describe this approach. The concept of Instruction Synthesis is that the sound is specified exclusively in terms of logical instructions [Roads [1996]. Instruction Synthesis was efficient and could be run in real-time on the most inexpensive microcomputers of the time. PILE operated in real-time and was capable of producing several layers of different sounds up to four channels. Instructions fell into the following categories [Berg 1979a]:

- Manipulation of the accumulator
- Manipulation of external devices
- Manipulation of variables
- Manipulation of lists
- Manipulation of program flow

The research background of PILE was ASP (Automated Sound Programs), a collection of 22 programs written by Berg in MACRO-15 which modeled number manipulation and temporal distribution systems (e.g. counting, calculating, comparing, choosing, repeating etc) to generate and manipulate binary data.

According to Paul Berg the user's task could be described [berg 2009, pp.83]:

> *A possible scenario for developing a composition was to develop a section by writing code, listening, and refining. The program was deterministic, such that given the same initial values, the same result would always be reproduced. Several sections could be developed independently and their code concatenated to create larger structures. Different sections could be variants of a previous section but with different initial*

conditions. The final result (a complete composition) would be produced in real time.

Since the acoustic qualities of the produced sounds may not always be predictable, the composer who uses instruction synthesis works in a trial-and-error mode [Roads 1996].

S.R.Holtzman designed and implemented in 1979 an Automated Digital Sound Synthesis Instrument in an attempt to provide a hierarchical organization control of Instruction Synthesis. He developed the Program Generator that generated text in compiled machine code, which synthesized the sounds [Holtzman 1979].

6.4 ***SAWDUST***

SAWDUST is a computer program conceived by Herbert Brün in the 1972 and implemented by Gary Grossman at the University of Illinois in 1976. SAWDUST was written in the C programming language under the UNIX operating system, running on a PDP 11/50. Jody Kravitz implemented a second expanded version in 1980.

Like Koenig's SSP, SAWDUST is also concerned with the compositional structuring of waveforms. In the program, the composer specifies waveform segments, called *elements*, which are then linked, merged, concatenated, repeated, and eventually interpolated, by the use of a limited number of operations [Brün *et al* 2001, pp.5]:

> *The computer program which I called SAWDUST allows me to work with the smallest parts of waveforms, to link them and to mingle or merge them with one another. Once composed, the links and mixtures are treated, by repetition, as periods, or by various degrees of continuous change, as passing moments of orientation in a process of transformations.*

These operations are carried by a number of functions like ELEMENT, ELIST, LINK, MINGLE or, MERGE. An interesting function, called VARY, gradually transforms one link into another by selecting polynomials (of degrees 3 to 7) that connect their elements in a number of steps or samples.

In contrast to SSP, the emphasis does not lie on a rule-based approach to composition, but rather on the extension and relocation of musical material to the waveform level. The composer is “forming sounds just as precisely as the macro events of his composition” [Brün 1969].

In SAWDUST, the user’s task could be described as [Brün *et al* 2001]:

- define a set of ELEMENTS (amplitude and time values)
- define a set of LINKS (sequences of elements)
- define a set of TRANSFORMATIONS between the links
- play and listen the defined transformations, links or silences in a user specified SEQUENCE

With SAWDUST Herbert Brün composed *Dust* (1976), *More Dust* (1977), *Dustiny* (1978), *A Mere Ripple* (1979), *U-TURN-TO* (1980) and, *i toLD You so!* (1981).

6.4.1 Wigout & TrikTraks

Arun Chandra designed in 1994 two different programs, *Wigout* and *TrickTracks*, both incorporating the synthesis paradigm of SAWDUST and extending it in various degrees. The main differences are as follows:

In *Wigout* a segment can be one of three types: 1) a ‘wiggle’, a sequence of samples at one amplitude; 2) a ‘twiggle’, a sequence of samples whose amplitudes have a linear rise to and fall from a specified peak; and 3) a ‘ciggle’, a sequence of samples whose amplitudes rise to a specified peak and return to

their starting magnitude in two second-order polynomial curves [Chandra 1994, Miranda 2002].

TrickTracks introduces the concepts of ‘standard’, ‘polynomial’ and AMFM paths. ‘Standard path’ refers to using a standard waveform (sine, square, triangle, or sawtooth) to control the path of the variables amplitude and number of samples. Polynomial paths were also used in SAWDUST in the implementation of the VARY algorithm. Chandra used them by specifying equally spaced zero-crossings, then scaling them to their specified limits. AMFM paths incorporated the Amplitude Modulation (AM) and Frequency Modulation (FM) algorithms.

7 Sonic Emergence

> *"the collision of hail or rain with hard surfaces, or the song of cicadas in a summer field. These sonic events are made out of thousands of isolated sounds; this multitude of sounds, seen as a totality, is a new sonic event. This mass event is articulated and forms a plastic mould of time…" [Xenakis 1992, pp.9]*

A significant phenomenon encountered in microsound systems is that of *emergence*. The notion of emergence is also widely used in the field of computer-generated art. It is influenced mainly by research in cognitive science, evolutionary biology, philosophy of science, cybernetics, systems theory, and artificial life. Despite its importance, there is no agreed definition of the notion of emergence. Different fields disciplines tend to use the term with relatively different meaning. McCormack [2004] and Whitelaw [2004] provide outlines of the use of the concept of emergence in scientific and technical literature.

The Oxford English Dictionary provides a rather general definition for emergence [Oxford 1989]: "The process of coming forth, issuing from concealment, obscurity, or confinement. Also said of the result of an evolutionary process."

We may define emergence as the way complex patterns and behavior arise out of dynamic interactions between agents in a system or environment.

Jeffrey Goldstein suggests that emergence is: "the arising of novel and coherent structures, patterns and properties during the process of self-organization in complex systems" [Goldstein 1999].

Stephanie Forrest writes from the perspective of the field of emergent computation [Forrest 1990, pp.8]:

> *"In these systems interesting global behavior emerges from many local interactions. When the emergent behavior is also a computation, we refer to the system as an emergent computation. ... Three important and overlapping themes that exhibit emergent computation are self-organization, collective phenomena, and cooperative behavior (absence of any centralized control)."*

In chapter (11) we will see specific computer models (stochastic, chaotic, fractals, Lindenmayer Systems, cellular automata) that are capable of generating emergent behavior and that are taking part in this research.

7.1 ***Hierarchical levels and emergence***

One way to comprehend emergence is to conceive a system as operating in different levels. We consider these levels ranging from local to global or otherwise from micro to macro. Joris Deguet and colleagues proposed that at least two levels required for a system to exhibit emergence behavior [Deguet *et al* 2005]. More than two levels make a hierarchy.

The concept of integrated levels or *strata* is used in ecology to describe the organization of a range of phenomena spanning from the micro-levels of non-biological realm (e.g. subatomic particles, atoms, molecules, macromolecules) through the various levels of biological phenomena, to the planetary macro-level (e.g. ecosphere). Alex Nivikov writes on the concept of integrated levels of a structural and dynamic hierarchy and the emergent properties in biological phenomena [Nivikov 1945, pp.209]:

> *"The concept of integrative levels of organization is a general description of the evolution of matter through successive and higher orders of complexity and integration. It views the development of matter… as continuous because it is never ending, and as discontinuous because it passes through a series of different levels of organization… In the continual evolution of matter new levels of complexity are superimposed on the individual units by the organization and integration of these units in to a single system. What were wholes on one level become parts on a higher one"*

The above concept is useful for the conception of an integrated hierarchy of structuring levels in generative art in general and the microsound synthesis of sound-objects through algorithmic means in particular. The concept of hierarchical organization of matter is in direct analogy to the hierarchy of time-scale levels in music composition as suggested by Curtis Roads and we have already seen in section (2.1).

Mitchell Whitelaw suggests from the perspective of artificial life and generative art that in general two levels can be distinguished. On the lowest end, the computational level (or a hierarchy of levels) of formal rules and interactions, and on the higher end, the level of complex behavior and emergence:

> *"This shared notion of emergence can be transcribed into a structural template made up of two levels: a local (computational) level, where complex interactions are driven by a set of formal rules, and a global level, where behaviors appear as patterns in time or space." [Whitelaw 2004, pp.214]*
>
> *"The computational level can be thought of more generally as a technological substrate, a designed framework of software and hardware. Similarly, the global emergent level can be thought of as the phenomenal and behavioral product of that technological substrate." [Whitelaw 2004, pp.214-215]*

7.2 ***The emergence of higher order sonorities***

We have already seen that microsound synthesis provides the conseptual and experimental framework of composing the sound instead of composing with sound. Regarding that in microsound the compositional activity operates in a level below any representation of the sound object, the composer is called to cope with the issue of emergence of 2nd order sonorities.

Agostino di Scipio questions on the connection between the structuring of microtemporal relationships among acoustic quanta and the emergence of the macrostructure in the form of a sound object [Di Scipio 1997]. Di Scipio regards that answering to this question is actually an assertion of a particular concept of sound modeling and therefore a compositional decision. ”Any answer to this question would require from the composer an ability to implement his/her own theory of sonological emergence” [Di Scipio 1997].

If we focus on sonological emergence on the level of sound object construction we can distinguish microsound operations focusing on at least two different hierarchical levels:

(1) the micro-level: this includes granular synthesis and other varieties of particle synthesis techniques

(2) the sample level: this includes non-standard synthesis techniques

These two hierarchical levels provide the framework for two distinct microsonic representations. On the one hand, microsound operations on the micro-level usually require some form of sound particle modeling which is further distinquished in a) individual particle formation (e.g. grains, pulses, etc) and, b) organization and sequencing of particles. Sound particles should be conceived as separate units that are constructed in an intermediate level and feature

specific sonological properties. On the other hand, non-standard operations on the sample level usually require direct sample patterning and waveform construction. In the latter case the composer works “in the single domain - the linear space of the sample sequence” [Di Scipio 1997]

Iannis Xenakis, years before the application of computers in microsound synthesis, suggests on the emergence of higher order sonorities from granular clustering and sequencing [Xenakis 1992, pp.47]:

> *“The basis of the timbre structures and transformations will have nothing in common with what has been known until now… Suppose that each point of these clusters represents not only a pure frequency and its satellite intensity, but an already present structure of elementary grains... We believe that in this way a sonority of a second, third, or higher order can be produced.”*

Agostino Di Scipio further suggests on non-standard synthesis that the modeling process whilst it utilizes microtemporal structuring actually targets the morphological emergent epiphenomenon [*di Scipio 1996, pp.67*]:

> *“Just as musical form can be understood as the epiphenomenon of a dynamical process captured in a model of musical design (i.e. at some macrotemporal scale), in computer music the properties of the sonic structure – whose local gestalt is usually called timbre – could themselves be understood as epiphenomenon of microtemporal compositional processes, unrelated to acoustic models but capable of modeling a phenomenon of morphological emerge”*

Albert S. Bregman in his cognitive analysis of auditory events used the notion of “*auditory stream”* to describe the distinctive perceptual identity of sound objects. For Bregman, a stream is perceptually formatted by (1) the obscurity of its constituent parts, and (2) the appearance of the emergent properties of the

stream as a united whole. For Bregman the grouping of acoustic information on a lower level give rise to the emergence of the auditory stream gestalt in the perceptual level [Bregman 1990, pp.138]:

> *"Treating a stream as a unit involves calculating its properties from only that set of acoustic features that has been asigned to it. Sometimes, in the study of perception, we speak about emergent features. These are global features that arise at a higher level when information at a lower level is grouped. "*

Composers by applying compositional techniques below the hierarchical level of the sound object actually work on a subsymbolic level. This is because the micro-sonic level of operation lacks the symbolic properties that emerge as an epiphenomenon on the higher level of listening experience. The network of relationships between particle units in the microlevel does not have any linear association with the perceived sound properties of the sound object and any syntactic feature experinced by the listener. This non-linearity, sometimes exhibited to the degree of non-association, makes rather difficult to describe the network of relationships between particle units in terms of a musical syntax. Thus we can regard that microsonic structuring operates in a presyntactic and subsymbolic level.

> *"Thus, according to these views, Physics presents an immense phase space of possibilities, in which it is impossible to determine exactly what will emerge at higher levels. Emergence can only be recognised after it has occurred, since it cannot be predicted in principle. emergence in computation is unrepresentable, in the sense of the product of elements interacting in ways that give rise to properties that cannot be predicted." [McCormac 2001, pp.6]*

Polanyi [34] recognised that while physics may be able to describe what is going on at a micro level, the macro emergent properties cannot be predicted

from the micro level physics, because they are computationally irreducible – determined by boundary conditions at the macro level. That is, the lower level laws are unspecific [9] with respect to the higher-level phenomena they may produce.

8 Computers, Cognition and Music Analysis

Concepts, methodologies and tools from the interdisciplinary field of cognitive science have been used for a long time both in the analysis of existing music as well as the creation of new music. Cognitive models consider intelligence as information system. Mental processes are regarded as effective computations that can be formulated by certain rules and representations. Cognitive musicology utilizes computer modeling and simulation to study music-related knowledge representation. Particularly, it studies how music is represented, stored, perceived, performed, and generated with the help of computers. It focuses more on the process of musical thinking than on products. Cognitive musicology uses concepts and advances of Artificial Intelligence and cognitive models are formalized, implemented and tested by empirical verification.

Otto Laske, a key researcher in cognitive approach to music composition, proposed the concept of music analysis by synthesis, thus simulating the generative process of music composition. Laske states on the use of computers in the formalization of music activity [Laske 1988, pp.67]:

> *"For the first time in the history of musical research, the computer program provides a medium for formulating theories of musical activity, whereas prior to its emergence only theories of musical artifacts had existed. As well, computer programs inevitably drew attention to the structure of musical decision-making in real time, thus to musical processes; they demonstrate the possibility of formulating musical knowledge in a procedural form, thereby highlighting the fact that musical knowledge in humans exists largely in that form."*

Concerning composition theory, Laske states that [Laske 1989, pp.119]:

> *"composition theory is a theory about the process that underlie the design and realization of musical compositions. In particular composition theory is about processes that are based on the use of explicit rule system."*

Analysis of non-standard synthesis, as part of computer aided algorithmic composition, can make particular use of methodologies of cognitive musicology. In this case, the methodology of cognitive musicology to model the composition process by computer simulation is identical to the methodology that the composer uses in the creative process itself. The computer is not only used for the final realization of sound, but it takes part in the processing of abstract signs and relationships. The rules of the composition are already "there", as part of the creative process. Computer aided algorithmic composition systems are capable of retaining information that in other compositional cases it is lost. G-M. Koenig states respectively [Koenig 1983, pp.31]:

> *"Although the composition of music - with or without computers – depends to a great extent on subjectively experienced criteria, the algorithmic description of the production process adds an objective feature; because of it, "form" is no longer the personal manner the musical material is presented or the listener's perception is guided rather, it is the rationally discernible, reproducible effect, detached from the composer, of an organized system imposed on arbitrary material"*

If we want to analyze the procedural aspects of a non-standard synthesis system, we can easily reconstruct the whole or parts of it. Since the compositional process of non-standard synthesis is computational itself, it makes cognitive musicology potentially a suitable approach for its analysis.

However, it should be clear that not every aspect of the compositional process could be successfully modeled by algorithmic processes. Even more, neither

compositional thought nor human thought in general could be totally modeled by cognitive science.

Any creative process, algorithmic or not, is always conditioned by human decisions and interpretation, either directly or by the mediation of other human-created algorithms. The creation of musical algorithms involves listening as an integral part of it. The composer listens to the musical results, possibly redefines the algorithm, refines the input parameters and takes compositional decisions that affect the final realization of the compositional process.

G-M. Koenig terms this process “aesthetic integration” and suggests by referring to algorithmic score generation system: “the algorithms embody a general idea of a piece, while the musical data only maintain abstract relations and support the realization of the score, which concretizes these relations.” [Koenig 1993].

Otto Laske himself admit that listening “is a mysterious process that is little understood, since it leaves no traces and encompasses perception as one of many ingredients... (Listening is) massively based on imagination, that is massively interpretive” [Laske 1993].

Additionally, Peter Kugel suggests, “musical thinking cannot be wholly accounted for in computational terms” [Kugel 1990]. This suggestion is based on Myhill’s Thesis which claims that although “all musical thinking can be characterized scientifically… certain aspects of musical thinking cannot be precisely characterized in terms of computations alone” [Kugel 1990]. Kugel considers Myhill’s Thesis as the aesthetic analogue of Gödel’s Incompleteness Theorem or of Church’s and Turing’s theorem about the undecidability of

Hilbert's Entscheidungsproblem. Furthermore, Kugel suggests that the evaluation of a composition as beautiful can never be achieved by computation, therefore the compositional process, like listening, involves more than computing [Kugel 1990].

In conclusion, there is always the necessity of human listening and interpretation in the operation of an algorithmic music system. Although, algorithmic composition in general and non-standard synthesis in particular, provides the cognitive sciences with the most possible formalized data for its analysis, there will always be “missing” compositional aspects that involves human interpretation and which cannot be modeled by the computer.

9 Algorithmic Models

9.1 *Algorithms and Musical Procedures*

The term algorithm does not have a generally accepted definition. There is an ongoing debate between researchers in formalizing the term. In an informal attempt, we could define algorithm as "a set of rules that must be followed when solving a particular problem" [Oxford 2006]. Computers require algorithms to process information. Computer programs, those written in imperative languages, specify an algorithm using declarations, expressions, and statements [Wilson 1993]. Algorithms vary from simple arithmetic operations to very complex procedures.

In the field of computer composition, algorithms control, transform and, generate some or all of the structural levels and parts of a musical composition [Essl 2003]. During these processes an algorithm may generate ordered or unordered musical events. Musical events may follow either dynamic or static time evolutions. In general, any information-processing model that generates or transforms data can potentially be applicable in the control of musical parameters and events thus becoming a musical algorithm. Nevertheless, none abstract process can be considered as musically neutral neither can be definably musical [Wooler et al 2005].

Algorithmic models can be either generative or transformational. This diversification takes the historical parallelism in the taxonomy of "classical analog studio" devices in three categories: sound generators, sound transformers, and mixers and recorders [Ciamaga 1975]. Generative

algorithms are formalized processes that produce musical events or parameter values. Transformational algorithms are formalized processes that permute existing values, thus they require to be supplied by data. Both generative and transformational algorithms may be controlled by various parameters. In turn these parameters can be controlled either by internal processes or by other external algorithms.

Algorithms, either generative or transformational, may encapsulate one another in more complex formalizations. The encapsulation of algorithms within algorithms result in a hierarchical tree where each junction is either a value or a procedure. Thus, both generative and transformational algorithms may contain sub-algorithmic processes of any complexity.

Algorithmic models are not definitely categorized as generative or transformational. For example, a stochastic algorithm is transformational when serving for the configuration of the spectral data of an existing sound. Further, a stochastic algorithm is generative when serving for the stochastic synthesis of a novel sound.

Algorithms can be applied onto the macrolevel, the mesolevel or the microlevel of sound composition. For example, the chaotic model of the logistic function can serve in the generation of a sequence of notes or it can be applied in the direct synthesis of a sound

9.2 *Permutations*

The permutation of musical material is one of the earliest algorithmic musical processes. Generally speaking, a permutation is a rearrangement of a list or a

set of musical objects or values in a particular order. Permutations are studied by the field of combinatorics. Permutations are usually simple algorithmic models. More complex approaches lead to the practice of composing permutations. A composition is defined by performing two or more permutations in succession.

Permutation algorithms usually contain the transformational processes of selection and mapping. Selection is the method of choosing one or more items from a list. Mapping is the method of transforming a value from one scale to another scale. The scaling function can be either linear or non-linear with various features.

Permutation algorithms can act on replacing or substituting either musical or extra-musical material and symbols by musical objects. One of the earliest approaches of algorithmic composition is the automated musical structure generation method by Guido of Arezzo, utilized in the turn of the 10^{th} century. In his work *Micrologus*, Guido of Arezzo describes a system for the automatic generation of melodies out of a text. In that system, letters, syllables and other components of a verse are replaced by musical notes and melodic phrases among other mapping procedures [Essl 2007].

The twelve-tone technique invented by of Arnold Schoenberg in 1921 is one of the most prominent examples for the application of permutations in the generation of musical material and major landmarks of 20th century musical thought. In the twelve-tone technique, arranging the twelve pitches of the chromatic scale in a particular sequence formed a “tone row”. All the harmonic and melodic material was systematically composed by permutating the original

pitch series. Basic operations for the permutation of the initial series is retrograde, inversion, transposition, augmentation and others. The next generation of composers (Boulez, Stockhausen, Nono etc) of the serial school extended the technique to other music material (such as duration or dynamics) even to all possible elements of music. The term, "'combinatorial' appears to have been first applied to twelve-tone music by Milton Babbitt in 1950 [Whittall 2008].

S.R.Holtzman use similar permutation algorithms in his Generative Grammar Definitional Language (GGDL) program. There he defines a set of “transformational rules” for: structural change indexes, inversion transformation, transposition transformation, retrograde transformation and merge transformation of musical material [Holtzman, S. R. 1980].

The SSP sound synthesis program by G.M.Koenig use algorithms called: expansion, reduction, reorder, isolation and copy for the selection, formation and permutation of list of data that eventually used in microsound synthesis. These algorithms make use of tendency masks for dynamic minimum and maximum value handling [Banks et al 1979]. A related algorithmic model is proposed by James Tenney [Tenney 1969]. Paul Berg’s AC Toolbox provides among others, the “low” and “high” arguments for data generation and control in a flexible computer music environment [Berg 2003].

The Sawdust microsound synthesis program by Herbert Brun uses the notion of elements as structural parts of a sound waveform. Within Sawdust, the “*link”* operation defines a sequence of elements while the “*mingle”* and “*merge”*

operations combining "*links".* The "*very"* operation is a complex algorithm utilizing polynomials that transforms an initial "link" to a final one.

9.3 *Stochastic*

Stochastic algorithms utilize processes derived from probability theory. In stochastic process, sometime called random processes, there is some degree of indeterminacy involved in the evolution of the generated events. The set of events is called "state space" while the set of parameters is called the "parameter space". Even if the initial condition is known, a stochastic algorithm may give many possible results, but some of them may be more probable then others. Thus stochastic algorithms are the counterparts of deterministic algorithms.

Probabilities are used to describe phenomena that are either too complex to describe in detail (molecular movement in a liquid) or exhibit an intrinsic stochastic behavior (elementary particles). In computer music, the interest in probabilities does not lie in describing but rather creating complex musical events or group of events. The interest may either lie in describing macroscopic features of a musical entity (e.g. masses of sounds) or the microscopic, individual characteristics of one or some of its members (e.g. the occurrence and the timbral characterostics of a particular sound).

Xenakis suggested as early as 1955 the replacement of deterministic causality of serial composition by stochastic processes. He proposed the statistical organization of musical events as the most appropriate representation for the manifestation of natural events. His composition *Pythoprakta* (1955-56) was the first that employed stochastic processes [Xenakis 1992]. At the same time

Lejaren Hiller developed a computer program that used random processes to compose the *Illiac Suite quartet* (1956-57) [Hiller et al 1958]. James Tenney's *Stochastic String Quartet* (1963) was inspired by both Xenakis and Hiller [Tenney 1988]. In 1971 G. M. Koenig developed the SSP computer program that pioneered the use of parametric boundaries or tendency masks for the control of random algorithms in sound synthesis [Koenig 1971]. Some years later Barry Truax incorporated also the concept of tendency masks in his POD granular synthesis system [Truax 1973].

Stochastic algorithmic models are implemented in various computer music systems throughout recent history, from Sever Tipei's MP1 [Tipei 1975[, David Zicarelli's Jam Factory [Zicarelli 1987] and Joel Chadabe M [Chadabe 1997], through Larry Polansky and David Rosenboom's HMSL [Polansky et al 1985], Heinrich Taube's Common Music [Taube 1989] and Eduardo Miranda's CAMUS 3D [McAlpine et al. 1999], to Paul Berg's AC Toolbox [Berg 2003] and Christopher Ariza's AthenaCL [Ariza 2005] among others.

Stochastic algorithms have been also used beyond the framework of creative computer music applications (mainly computer-assisted composition and machine improvisation with human performers) to music information retrieval, stylistic analysis and imitation of music as well as to cognitive modeling of music perception.

9.3.1 Probabilities

Probability is the chance that a particular event will occur. A probability is a fraction of a sample space, the set of all possible values the events may

acquire. Probability is usually expressed in a scale between impossibility and certainty. This scale ranges linearly between 0 and 1.

Probability distribution is a function that describes the probability of a random variable taking specific values. Usually probability distributions are visualized in a two-dimensional graph, called histogram. There are two general types of probability distributions: discrete and continuous. In continuous distributions a random variable can take a continuous range of value. The opposed happens to a discrete distribution, where a random variable can take values from a set countable of possible values [Goldberg 1986].

Frequently used discrete distributions are the Bernoulli, Bean Machine, Discrete, Multinomial, Binomial, Geometric, Hypergeometric, Poisson, Lattice and Zipf distribution [Goldberg 1986].

Frequently used continuous distributions are the Beta, Gaussian, Binormal, Cauchy, Half-Normal, Rayleigh, Laplace, Rice, Lévy, Logarithmic, Standard Normal, Logistic, Student's t, Exponential, Uniform, Gamma and Weibull distribution [Goldberg 1986].

Stochastic distributions are relatively easy to implement in a computer algorithm. Probability algorithms are used in computer music for the control of various levels of composition. For example stochastic algorithms can control the occurrence of events, which may be musical phrases, notes or sounds. They can also control various parameters of musical events such as frequencies, dynamics or various timbral aspects. Stochastic algorithms are utilized for the higher level control in microsound synthesis for the formation and control of the

various microevents (usually sound grains) that constitute complex sound events [Roads 2001].

Xenakis was one of the first composers that used stochastic algorithms for the composition of both instrumental and electronic works. Among other, he used Stochastic algorithms in the concept of granular synthesis as well as in the sequencing of microsound events in his GENDY computer program [Xenakis 1992].

9.3.2 Random Walk

The random walk algorithm generates a trajectory that consists of taking successive random steps [Feller 1968].

Random walks eventually tend to take extremely high values. These extreme values either exceed a meaningful compositional musical range (e.g. a range of frequencies or dynamics), either exceed some human perception levels (e.g. the audible range of frequencies) or exceed the computer's numerical representation limits (e.g the amplitude bit-depth). Therefore there is the need to limit the range of the random walk between some *boundaries*. These boundaries are often called *barriers*. Barriers can be reflective, thus bouncing the exceeding value back to the allowed phase space as a mirror, or wrapping, thus encountering topologically the phase space as a torus.

The step of the random walk can be (1) a constant value, (2) the output of a stochastic distribution, or (3) the output of a secondary random walk. In the latter case we can have nested random walks of various levels. The range of the possible step values can be controlled over time by tendency masks.

The elastic barriers can (1) remain static or (2) their ranges can vary over time by the help of tendency masks or other functions.

Random walks can take place in one, two or even higher dimensions according to the musical parameter space. Usually, one musical parameter corresponds to one dimension of the musical space.

Another name used for such an algorithm is Brownian Motion, which describes the movement of particles suspended in a fluid. Random walk can be considered as a simple form of Markov Chain that will be discussed in the next section.

Iannis Xenakis is the first composer that employed random walks in music composition. He states on the preface to the score of N'Shima:

> *"The melodic patterns of N'Shima are drawn from a computer-plotted graph as a result of Brownian movement (random walk) theory that I introduced into sound synthesis with the computer in the pressure versus time domain*" [Xenakis 1976, pp.1].

Xenakis used random walks in the following instrumental pieces: Cendrées (1973), Phlegra (1975), Theraps (1976), Retours-Windungen (1976), Epeï (1976), Akanthos (1977), Jonchaies (1977), Ikhoor (1978), Dikhthas (1979), Palimpsest (1979), Anémoessa (1979), Mists (1981), Komboï (1981), Chant de soleils (1983), Tetras (1983), and Thelleïn (1984). [Solomos 2001].

Random walks play an essential role in *stochastic waveform synthesis*. In this synthesis technique, Xenakis utilized two separate nested random walks for the amplitude and the duration of elementary parts of the generated sound waveform [Hoffman 2000]. An initial approach of *stochastic waveform synthesis*

was used in some parts of *La Légende d'Eer* (1977). With the advent of the GENDY program he composed *GENDY3* (1991) and *S.709* (1994). Stochastic waveform synthesis and the GENDY system will be discussed in section (13.1).

9.3.3 Markov Chain

The history of Markov models is well documented in the mathematical and scientific literature [Norris 1998, Bermaud 1999]. In Markov chain algorithms, named for Andrey Markov, one or more past events are used for the probabilistic calculation of the transition to the next state. The transition from one state to the next is in sequential or chainlike manner. The Markov chain can be represented by a transition matrix that describes the transition probabilities. Higher-order Markov chains take into account more than one past events for the transition probabilities. In this case the order of the Markov chain indicates the number of the past events.

A Markov chain can exhibit various properties. An event that can be followed by another event is regarded as *accessible.* If two events are bidirectionaly accessible from one to another are regarded to *communicate.* The communication between two events can have three possible properties: (1) It is *reflexive* if one event communicates with itself. (2) It is *symmetric* if two separate events communicate with one another. (3) It is *transitive* if one event communicates with a second event and the second event communicates with the first event. Group of events that communicate between them form *equivalence classes*. In this case only one event from one class can communicate with another event from a different class. *Recurring* are the events that are certain that will occur again after one or more transitions. On the

other hand, *transient* are the events that are certain that will not occur. A communicating class is *closed* if the probability of leaving the class is zero.

Charles Ames [Ames 1983, 1987, 1989, 1990], Denis Lorrain [Lorrain 1989] and Kevin Jones [Jones 1981] provided a thorough survey and examination of probabilistic and Markov chain applications in automated music composition.

Lajaren Hiller and Leonard Isaacson applied Markov chain algorithms of variable orders for the automatic generation of “experiment four” in their *Illiac Suit*e *quartet* (1956-57) [Hiller et al 1958].

Iannis Xenakis in his composition *Analogiques A et B* (1958-59) for nine string instruments and tape used Markov models to control the frequencies, dynamics and densities of musical events.

Recently, Eduardo Miranda and Adolfo Maia Junior developed a model for granular microsound synthesis using Markov Chains for the control of the evolution of sound in time and Fuzzy Sets for the definition of the internal structure of the sound grains [Miranda et al 2005].

9.4 *Chaos & Fractals*

Fractals (otherwise, *self-similar systems*) and Chaos (otherwise *non-linear dynamical systems*) provided a new paradigm in the understanding and perception of nature and have been used for the explanation of a multitude of diverse physical phenomena. Subsequently, fractal and chaos have been used in modeling various systems.

A diversity of fields has taken advantages on this subject: architecture, art, astrophysics, biology, chemistry, communications, computing, data

compression, economics, electronics, fluid dynamics, geology, geophysics, linguistics, meteorology, music, physics and signal processing.

Because of the complexity and the required number of calculations, the experimentation with such systems became only possible but with the advent of fast computing machines. The scientific work of Edward N.Lorenz and Benoit Mandelbrot was eminent for the development of the field.

As Peter Beyls states, “complex dynamic systems are an alternative to the constructivist approach to composition, i.e., the critical assembly of architectures of time according to some explicit scenario” [Beyls 1991].

9.4.1 Chaos

Chaotic is any system that at particular states it is very difficult to predict and at first glance it might be perceived as random. Chaos Theory first introduced by the mathematician Henri Poincaré at the beginning of the 20th century. Poincare proved that “It may happen that small differences in the initial conditions produce very great ones in the final phenomena. A small error in the former will produce an enormous error in the latter. Prediction becomes impossible”. Two identical chaotic systems, set in motion with slightly different initial conditions, can quickly produce very different results [Peitgen et al 1992].

The mathematical equations that exhibit chaotic behavior are relatively simple and can be easily implemented in computer algorithms. The basic feature of chaotic algorithms is *iteration*. The solutions produced from a single iteration are fed back and turn into input variables for the next iteration recursively [Flake 1998].

The sequence of values generated by a chaotic process is called an *orbit.* Since different initial conditions may produce different results, an orbit is representative for a particular algorithm only for that particular set of initial conditions. Subsequently, since there are infinite possible initial conditions there are also infinite possible orbits obtainable by a chaotic algorithm. Usually chaotic algorithms generate orbits within a limited range of values that define an n-dimensional *phase space*. The chaotic behavior of an algorithm traces orbits within the phase space without returning to the exact same position twice [Peitgen et al 1992].

Computers are limited to finite accuracy arithmetic. In other words, computer accuracy to represent a real number is limited to a particular length of decimal points. Accordingly, algorithms that are susceptible to the slightest differentiation of their initial conditions are subjected to accumulative errors during the iterative process. Although the global behavior of chaotic algorithms is manifested through computer representation, the detailed generated orbits differ due to system implementations [Peitgen et al 1992].

Chaotic systems can be classified into two categories according to the evolution of their manifestation over time: dissipative systems and conservative systems. *Dissipative systems* are those for which, after an initial transient phase, the volume of phase space decreases over the course of time. This decrease eventually establishes an *attractor* where all nearby orbits converges to it. Attractors can be (1) *fixed-point* with a particular value (2) *limit-cycle*, where an orbit oscillates periodically between a sequence of values, or (3) *strange,* where the orbit exhibits very complex behavior and typically have non-integer phase space and thus fractal structure. *Conservative systems* maintain a constant

phase space throughout calculations without exhibiting a transient phase not establishing any attractor. Nonetheless, conservative systems may exhibit distinctive behavior similar to attractors [Peitgen et al 1988].

In some chaotic systems time evolves in *discrete* steps. Theses chaotic systems (also called *iterated maps*) are comparable to simple feedback systems. Discrete chaotic systems that are applicable in the computer music context are: Arnold's cat, Baker's, Circle, Complex quadratic, Complex squaring, Duffing, Dyadic, Exponential, Gauss, Gingerbreadman, Hénon, Horseshoe, Ikeda, Logistic, and Tinkerbell map. In other chaotic systems, time is considered to be a *continuous* quantity, and the system is expressed as a differential equation. These systems generate orbits with smooth trajectories (comparable to the discrete systems where orbits are usually characterized by abrupt jumps). In the computer environment continuous time is a theoretical concept. In this case, reformulating the differential equations into difference equations using a process called integration simulates time continuity. Continuous chaotic systems that are applicable in the computer music context are: Lorenz attractor, Duffing equation, Rabinovich-Fabrikant equations, and Rössler map.

Chaos algorithms have been researched for the generation of various aspects of music structure. Jeff Pressing worked largely with the logistic map and explored its “quasi-chaotic behavior” for the creation of note sequences [Pressing 1988]. Michel Cogins investigated an algorithm that stochastically switches between different iterated function systems [Cogins 1991]. Rick Bidlack demonstrated four different algorithms and researched its application on pitch, dynamics, rhythm and instrumentation between note events [Bidlack 1992]. David Clark Little applied five chaotic algorithms (Lorenz, Verhulst,

Hénon, Barry Martin, and the Baker) in various aspects of score generation (designing melodic curve, defining meter, planning instrumentation, manipulating symbols, creating ornamentation and elaboration) and composition [Little 1993]. Gary Lee Nelson devised real-time computer programs where he implemented the logistic and the Verhulst maps. He generated the musical notes for his compositions *The Voyage of the Golah Iota* (1993) and *Colony* (1994) by controlling the algorithms in a dynamic manner [Nelson 1994]. James Harley introduced specific techniques for constructing dynamical algorithms along with his CHAOTICS software [Harley 1994, 1995].

Chaotic algorithms are also explored in physical modeling sound synthesis. Leon O. Chua introduced in 1983 a simple electronic circuit that exhibits non-linear behavior. Chua's circuit is easily implemented as a computer algorithm. The chua's circuit produces a chaotic attractor, known as "The Double Scroll". Because of its simplicity in implementation and its accurate theoretical model, it is considered elementary in the research of chaos theory and behavior. A number of researchers studied the chaotic behavior of the circuit in the context of physical modeling of acoustical instruments like single-reed and double-reed winds, brass, flutes, and bowed strings [Roded 1993, Gottfried et al 1993]. Billota and colleagues carried research on the musical implementations of Chua's circuit both in waveform generation as well as musical phrase composition [Bilotta et al 2005]. James N. Sears utilized Chua's circuit as part of the sound design for the play *Marisol* (2001). J.P. Mackenzie utilized the Lorenz attractor for the physical modeling of acoustical instruments (the tuba and the gong), a natural phenomenon (the sound of the wind), and a machinery (the rumble of a ventilation fan) [Mackenzie 1995].

Chaotic algorithms were also explored in the research of novel sonorities. Barry Truax explored the logistic and the quadratic maps for the higher-level control of granular synthesis [Truax 1990]. Richard Dobson and John Fitch utilized a modified version of the Mandelbrot set for direct synthesis and the generation sounds, control functions or amplitude envelopes [Dobson et al 1995]. Insook Choi investigated musically the chaotic processes of the Chua's circuit in an interactive real-time environment [Choi 1999]. Manzolli and colleagues developed the FracWave method for the interactive real-time control of a set of two-variable iterations, which are variations of the so-called standard map [Manzoli et al 2000]. Agostino Di Scipio is a prominent researcher and composer in the applications of chaos algorithms in direct synthesis of sound. He investigated extensively various iterative function systems like the sine map or the logistic map and explored their sonic properties in an non-standard synthesis approach. [Di Scipio 1994, 1996, 2000, 2002a]. During our research we have explored the non-linear behavior of two iterative cross-coupled digital oscillators in an interactive real-time environment and applied it in the context of a musical performance [Valsamakis et al 2005]. Georg Essl explored the chaotic properties of simple maps, in specific ranges that produce sonorities from chaotic, noisy-sounding responses, to pure and mixed sine wave, amplitude modulation, and pitch bending [Essl 2006].

9.4.2 Fractals

Fractals are objects that exhibit self-similarity on all scales of magnification. Particular patterns reoccur identically or similarly on different orders of magnitude. Fractal structures can be found everywhere in nature or can be

generated algorithmically on a computer. [Peitgen 1988]. Usually fractals are displayed graphically. The term *fractal* was first introduced by Benoit Mandelbrot:

> *"I coined fractal from the Latin adjective fractus. The corresponding Latin verb frangere means to "break": to create irregular fragments. It is therefore sensible – and how appropriate for our needs! – that, in addition to "fragmented" (as in fraction or refraction", fractus should also mean irregular, both meanings being preserved in fragment"* [Mandelbrot 1982, pp.4].

Fractals can be classified according to the degree and type of self-similarity. Objects that exhibit *exact self-similarity* on all levels are called *regular fractals*. This rigid property is found only in abstractly conceived objects. However, only some mathematical fractals, like the Sierpinsky gasket or the Koch snowflake, are similar everywhere and infinitely, no matter where we zoom in. Other mathematical fractals, like the Mandelbrod Set, exhibit *quasi self-similarity* and appear approximately identical at different scales. Exact and quasi self-similar fractals are *deterministic*. On the other hand, fractals that contain a statistical element and exhibit statistical self-similarity are known as *random fractals.* Random fractals describe natural objects and processes [Addison 1996]

There are four categories of fractals according to the generation technique they acquire: (1) *Escape-time* fractals defined by a formula at each point in a space. Examples of this category are the Mandelbrod set, the Julia set and the Lyapunov fractal. (2) *Iterated function* fractals that are produced by a fixed replacement rule. Examples are the Cantor set, the Sierpinski carpet, the Sierpinski gasket, the Peano curve, and the Koch snowflake. (3) *Random* fractals generated by stochastic processes. Examples are the Brownian motion,

the Lévy flight, fractal landscapes and the Brownian tree. (4) *Strange attractors* generated by chaotic models.

Fractals exhibit usually non-integer dimensions, called the *fractal dimensions*. There are many specific definitions of fractal dimension. Fractal dimentions are usually greater than the topological dimension of the generated object and less than its Euklidean dimension.

Larry Austin composition *Canadian Coastlines* (1981), utilizes explicitly the tracings of the natural fractal outlines of Canadian coasts to choose musical parameters such as pitch, rhythm, timbre, and duration [Dodge et al 1997]. Charles Dodge interpreted the Mandelbrod set for the construction of the tape part of his composition *Viola Elegy* (1987) for viola and computer processing. Additionally, in his algorithmic tape piece *Profile* (1984) all the compositional elements (timing, pitch and dynamics) were made by systematic application of 1/*f* fractional noise [Dodge 1988]. Garry Lee Nelson used fractal algorithms for score generation in his microtonal composition *Fractal Mountains* (1988-89) [Nelson 1996]. Charles Wuorinen has composed a number of works that employ 1/*f* fractal relationships including *Bamboula Squared* (1984) for orchestra and tape. Rolf Wallin used fractals in his percussion work *Stonewave* (1990) [Wallin 1989]. Michael McNabb's interactive composition for live-electronics *The far and Brilliant Night* (1990) uses real-time algorithms to adjust resonant comb filter and produce fractal melodies [McNab 1990]. Michel Cogins presented in 1991 his Iterated Functions System (IFS), which can produce, connected or disconnectedfractals, self-similar or non-self-similar fractals, and fractals that appear to contain non-fractal elements [Cogins 1991]. Brian Evans explored the Mandelbrod set in the csound programming environment [Evans

2000]. From 1995 onwards, fractal algorithms has been used on various levels of sound and music composition.

In the context of non-standard synthesis, Shahrokh David Yadegari used fractal algorithms for direct waveform generation using the concept of midpoint subdivision [Yadegari 1991, 1992]. In 1995, Gordon Monro proposed another non-standard direct synthesis method, called fractal waveform interpolation synthesis. Monro utilized a deterministic, iterated function fractal model. The composition *Dry Rivers (1995)* was produced entirely by the utilization if this method [Monro 1995].

9.5 *Grammars*

The Oxford Dictionary defines grammar as:

> *"the whole system and structure of a language or of languages in general, usually taken as consisting of syntax and morphology (including inflections) and sometimes also phonology and semantics*." [Oxford Dictionaries Online]

Subsequently the Oxford Dictionary defines syntax as:

> *"the arrangement of words and phrases to create well-formed sentences in a language".*

and explicating more:

> *"A set of rules for or an analysis of this"*

Analogously, music composition, irrespective based on notes or on sound objects, can be seen as having its own grammar and consisting of its own morphology, syntax and rules. The hierarchical structure of language can also

finds its analogy to the various parts and levels of a musical composition. The formalization of the hierarchical syntactic relationships within a structure with the help of abstracted notation defines a *formal grammar.* The linguist Noam Chomsky first introduced the concept of formal grammars in the late 1950s in his revolutionary book *Syntactic Structure* [Chomsky 1957].

There is a long debate between scholars whether music is a language or not. Over this debate we are commenting the words of Noam Chomsky quoted by Curtis Roads: "it all depends on one's definitions, and ultimately it is an unnecessary question; one shouldn't be diverted by it. I take this as the starting point..." [Roads et al 1979].

Grammars have been applied both for music analysis and composition of music.

A generative grammar can be defined formally by a four-element structure (N, T, P, S) where: N is a set of non-terminals, T is a set of terminals, P is a set of production rules, and S is the root token. A terminal is a token that it cannot decomposed. A non-terminal is a token that can be decomposed to other tokens, either terminal or non-terminal [Miranda 2001].

Noam Chomsky categorized grammars into types according to the level of restriction for the application of their rules. Additionally, the type is related to the generative capacity of the grammar. The higher the grammar's order the more restrictive and the more expressive it is. On the other hand, the lower the grammar's order the less control it has over the generation process and subsequently, a higher generative capacity. Chomsky proposed four grammar types [Chomsky 1957]: (1) Type-0 or unrestricted: imposes no restrictions on

the form of production rules and thus is not suitable for music composition. (2) Type-1 or context sensitive: is capable of generating all languages that can be recognized by a linear bounded automaton, a restricted form of nondeterministic Turing machine. (3) Type-2 or context-free are capable for generating multi-level hierarchical trees which is common in (most of the) music, natural languages and programming languages. (4) Type-3 or regular grammars: is capable of generating all languages that can be decided by a finite state automaton. Curtis Roads added two more types: (5) transformation and (6) regulated grammars [Roads et al 1979].

Steven Holtzman proposes five types of production rules: (1) Random selection by employing a set of possibilities. (2) Serial selection (or blocked generation) where tokens are selected randomly and all tokens are used before a single token is selected again. (3) Finite-state generation by the application of a transition matrix where tokens are selected stochastically according to the previous selected token. (4) Higher-level control (or non-system rewrite control), where tokens are indexed and selected by external functions of any complexity. (5) Meta-production rules, where a new set of production rules is generated automatically from two primary defined sets of rules [Holtzman [1981].

Bol Processor, created by Bernard Bel and James Kippen is a real-time improvisation and composition program that constructs a generating grammar by analyzing input musical data. Bol Processor is capable of modeling various music styles including Western classical music, serial music, minimalism contemporary music, and Indian classical music [Bel 1998, 2006].

EMI (Experiments in Musical Intelligence), created by David Cope, is considered as one of the most successful systems employing grammars in music composition. EMI is capable of generating music that is highly faithful to the style of the provided music. EMI uses three basic levels of operations: (1) analysis and deconstruction of input music into separate parts, (2) preservation of the most significant parts that indicate the particular style, (3) recombination into new compositions [Cope 1991, 1996, 2000].

One of the most radical approaches in music composition with grammars is Steven Holtzman's "automated non-standard digital sound synthesis instrument". In this approach, a grammar that specifies the rules for ordering machine instructions generates directly the waveform of the sound. These basic instructions include: store, retrieve, add, subtract, multiply, divide, logical shift etc. The generated sound is conceived as the direct monitoring of the operations of the computer. Holtzman composed the tape work *Machine Music ICL 2970* (1978) in which every possible operation executed by the ICL2970 mainframe is employed.

9.6 *Lindenmayer Systems*

An Lindenmayer system or L-system is a string-rewriting algorithmic system mainly used to formally describe the growth processes of plant development. L-systems were introduced in 1968 by biologist Aristid Lindenmayer [Lindenmayer 1968].

In rewriting systems, sub-terms of a formula is replaced with other terms. Whereas L-systems are closely related to Chomsky generated grammars, one main difference is that they perform string-rewriting in parallel instead of the

serial-rewriting mechanism of the latter. Additionally, due to their recursive logic L-systems are capable of exhibiting self-similarity and thereby generating fractal forms.

L-systems are used mainly as algorithmic implementations of the morphogenic theory of *emergence*. Emergence is a formal growth process by which “a collection of interacting units acquires qualitatively new properties that cannot be reduced to a simple superposition of individual contributions” [Prusinkiewicz et al 1996].

In L-systems, strings represent the current state of the modeling structure at each turn. Strings usually consist with alphabet letters. The initial string is called *the axiom* of the system. A set of *rules, called productions,* determines the way the system rewrites specific symbols. Rules describe the substitution of a *predecessor* symbol by a *successor* string. During the process, if a symbol in the current string matches a predecessor, it is replaced by the corresponding successor in the output string. On the other hand, if a symbol does not match with any predecessor symbol it is copied as it is in the output string. The output string is then fed back into the system and becomes an input string. Since the length of the string eventually approach an infinite limit, it necessary to constrain the number of iterations in order to allocate computational time and storage space.

According to the type of grammar in use and the method the production rules are applied we can have various types of L-systems: context-free or context-sensitive, deterministic or non-deterministic, bracketed, propagative or non-propagative, with tables, parametric, with extensions [Manousakis 2006].

Przemyslaw Prusinkiewicz proposed in 1986 one potential strategy for composing music using L-systems by performing spatial mapping of pitch height and amplitude [Prusinkiewicz, 1986]. Garry Lee Nelson at that time was highly influenced by Prusinkiewicz. He extended and applied L-systems in his compositions *Summer Song* (1991) and *Goss* (1993) [Nelson 1996].

Jon McCormack explored extensively L-systems in the musical context. He compared L-systems as successful alternatives of Nth-order Markov models, finite state automata and Petri nets. In one of his applications, he used non-deterministic, context sensitive grammars for the generation of complex music compositions from relatively simple rules. [McCormack 1996].

Roger Luke DuBois explored a variety of composing methodologies with the application of L-systems. He experimented with various mapping taxonomies of mapping the output of L-systems to music material. His research was focused mainly on real-time processing of live musical input [DuBois 2003].

Mapping strategies were also discussed by Francis and Jacques Soddell [Soddel et al 2005], Worth and Stepney [Worth et al 2005] and Nigel Morgan [Morgan 2007]. Peter Beyls created a modular Lisp environment where L-systems (among other algorithms) can be used for real-time melody generation and harmonization [Beyls 2000]. Pedro Pestana developed an interactive application based in L-systems for real-time improvisation with a computer [Pastena 2009].

Michael Cogins demonstrated a context-free L-system based on a group of operations that abstracts and extends neo-Riemannian transformations. This L-system is capable of generating a number of simultaneous musical voices that

lead from one chord to another and finally define a complete score [Cogins 2009].

Lourenco and colleagues proposed the use of genetic algorithms to change the set of productions rules between successive iterations to increase variability [Lourenco et al 2009].

The composer Hanspeter Kyburz experimented with L-Systems for the automated organization of small motifs in his work *Cells*, for saxophone and ensemble (1993–1994) [Supper 2001]. Michael Edwards in his composition *Tramontana* for viola and computer (2004) employs transitioning L-Systems with the help of Fibonacci-based folding structures [Edwards 2009].

Stelios Manousakis developed a computer music program that employs L-systems and is capable in generating musical structures from the macro-level down to the sample level. In this context, Manousakis proposed a novel non-standard synthesis method for direct waveform construction and experimented with various control methods [Manousakis 2006, 2009].

9.7 *Cellular Automata*

The concept of Cellular Automata (CA) introduced by Stanislaw Ulam and John von Neumann in their research on the process of reproduction and growth of form [Burks 1970].

CA are dynamic systems that produce global self-organizing behaviour emerging from the interactions of local elementary units. In CA time and space are discrete. CA are of any finite number of dimensions but usually are implemented either as a 1-dimensional linear array or as a 2-dimensional greed

array. Greed boundary conditions are either fixed or wrapped around (forming a torus). CA evolves simultaneously over time by an algorithmic process that operates in parallel on every cell of the array. A local transition rule specifies the new state of each cell. Rules are usually taking into account the current state of the cell as well as the state of the cells in its nearest neighborhood. The subsequent states of a CA are called *generations.* The starting state of the CA, called the *seed* or *initial conditions*, is either randomly generated or a single cell in the centre may be take a particular value [Wolfram 2002].

Usually, the cells in CA can take a discrete number of states. However, there is a special category of CA in which the states of the cells are *continuous*, usually in the real number range [0, 1]. Thus, Continuous CA cells can take an infinite number of possible states [Wolfram 2002].

The behaviour of CA depends both on their rules and the starting state. Stephen Wolfram classified CA according to their behavior in terms of complexity. According to Wolfram there are four general classes of CA [Wolfram 1984]:

- Class 1: Patterns evolve into a stable, homogeneous state or disappear.
- Class 2: Patterns evolve into oscillating structures.
- Class 3: Patterns evolve chaotically and never repeat.
- Class 4: Patterns evolve into complex structures. Eventually may exhibit class 1 or 2 behaviour, but this occurs usually after a large number of steps.

Christopher Langton called the undoubtedly interesting but relatively rare behaviour of class 4 as “computation at the edge of chaos” [Langton 1991]. Langton also proposed the λ parameter as a phase transition indicator between

ordered and chaotic behavioral regimes [Langton 1991]. Although the λ parameter appears quite useful for the control of the behaviour of CA it should be used with caution. Other researchers as well as Langton himself consider it as not always being able to work correctly.

Iannis Xenakis was the firs who applied CA in instrumental music composition in his work *Horos* (1986) where the state of the cells controlled the occurrence of the musical events [Solomos 2005].

Peter Beyls started his thorough research and experiments in CA for algorithmic composition and performance during the 1980s. He developed the first MIDI music system with CA. In his work proposed various innovations like new types of CA, the use of time dependent rules that change over time, history tracking of selected previous generations, investigation of a small network of interconnected 2D CA, the use of Langton's λ parameter in a real-time environment as well as various mapping strategies [Beyls 1989, 1990, 1991, 1997, 1998, 2000].

Computer music systems that employ CA to generate and control MIDI music data are: the *Cellular Automata Music* (CAM) by Dale Millen [Millen 1990], the *Cellular Automata Workstation* by Hunt, Kirk and Orton [Hunt, et al 1991], The *CAMUS* and *CAMUS 3D* by Eduardo Miranda [Miranda 2003, Dewdney 1989], the *Reaction-Diffusion (R-D)* system by Andrew Martin [Martin 1996], the *FractMus* algorithmic composition system [FractMus 2005], the *Softstep*, *MusicWonk* and *Artwonk* commercial modular algorithmic development applications by Algorithmic Arts [Dunn 2006], and *Harmony Seeker* by Eleonora Bilotta and her colleagues [Bilotta & Pantano 2002] among others. The majority

of these systems have focused on mapping CA values to pitch and duration [*Burraston* et al 2004].

A number of computer music applications use CA to generate and control a microsound granular synthesis engine. Peter Bowcott utilized the 2D Life CA for the automated generation of Csound data [Bowcott 1989]. The *Cellular Automata Workstation (CAW)* controlled tendency masks for the synthesis engine [Hunt, et al 1991]. *Chaosynth* by Eduardo Miranda used the model of the neural reverbatory circuit to drive a bank of oscillators [Miranda 2003].

Christopher Ariza proposed implementations beyond any “pure” concept by bending or even breaking the rules of CA. He introduced the term *automata bending* analogously to the electronic circuit-bending techniques demonstrated by Ghazala and others [Ghazala 2004; Collins 2006]. Particularly, he proposed two types of automata bending, random cell-state mutation and dynamic probabilistic rule-sets, concepts derived from the field of evolutionary computing. With automata bending “CA rules that produce unexceptional behavior can be invigorated and made into more useful generators” [Ariza 2007].

A non-standard synthesis approach for direct waveform generation is the *Linear Automata Synthesis (LASy)* system by Jacques Chareyron [Chareyron 1988, 1990]. LASy employs a 1D Cellular Automaton to represent a wavetable. Individual cell values correspond to audio sample values. After every playback cycle, the state of every cell is computed according to the CA transition rule, thus the automaton produces a self-modifying waveform. LASy produces a

variety of sonorities according to the CA transition rules. The sound synthesis model is very close to the Karpus-Strong algorithm [Karpus *et al* 1983].

9.8 ***Sound Modeling***

We can formally define a sound synthesis model as an appropriate computer algorithm with a finite set of control parameters. The execution of the computer algorithm takes the set of parameter data and generates the bulk of sound data that represent the temporal structure of a sound.

Usually, the size of the parameter data is smaller than the data of the generated sound. Moreover, there are algorithms that with few control parameters are capable of producing sounds with very complex temporality. In addition, there are algorithms that are capable of generating, with the use of a finite set of instructions, sounds with infinite duration. Viewing this from the perspective of information theory, a sound synthesis model is a data reduction method in the representation of a sound.

A computer algorithm, that formally describes a sound model, can be considered as the algorithmic reduction of the generated sound. The execution of the algorithm unfolds its inner logic into sound. The logic within the algorithm is imprinted into the sound. However, there is no direct linking between the complexity of the algorithm and the perceived complexity of the generated sound. There are algorithms with simple syntax capable of producing sounds with very complex morphologies. For example algorithms consisting of a single iterative process can produce very complex sonic results. In contrast, algorithms with more complex syntax are producing more or less only monotonous sounds. This is the case of Parallel Frequency Modulation.

The sonic results of a sound model are dependant by both the computer algorithm and the parameter data. A very sophisticated computer algorithm with some parameter data may generate very monotonous sounds or no perceivable sound at all. In contrast, a rather crude algorithm with thoroughly chosen parameter data may produce more interesting sonic results.

A sound model is a generalization of the sound phenomenon that tries to represent. The tradeoff for the data reduction advantage is the incapability of generating all possible sounds with a single sound model. Sound models are capable of producing only a limited range of sounds. Moreover, each sound model usually generates sounds with distinctive and characteristic morphologies, hence limiting the listening impression. Each synthesis model has its own characteristic mark.

Some sound models are capable of linking directly the choice of the parameter data to the sonic result, thus producing definite predictable sounds by the musician. For example, using simple Frequency Modulation with sinusoids, one can achieve a totally predictable sound spectrum knowing the mathematical basis of the method. These models help the composer in making conscious decisions during synthesis. On the other hand, other sound models do not rely on a closed theory and the sounds that are generating are much less predictable. For example, this is the case of feedback Frequency Modulation or the non-standard approach in sound modeling. In these cases the composer needs to approach empirically the synthesis model and experiment intuitively with it.

One can classify the models of sound according to their object of modeling [Smith 1991]. There are three approaches that model sound through a synthesis algorithm, according to:

- the physical source of the sound
- the acoustics of the sound
- something independent of both representations

These three approaches are:

- Physical Modeling
- Acoustic Modeling
- Abstract Modeling

A fourth model of sound that cannot be viewed as a sound synthesis method but rather as a method of sound representation or as a method of sound transformation is:

- the digital representation of the sound itself

This approach is:

- Sampling.

If one would classify Sampling as a synthesis method, then every single sample of the digital representation of the sound should be considered as a separate control parameter. In this view, sound modeling through Sampling involves no data compression.

Following the above taxonomy of sound synthesis algorithms proposed by Julius O. Smith, non-standard synthesis should be placed as a sub-category of

abstract modeling [Smith 1991]. Although Smith does not make any direct reference to non-standard synthesis, it is obvious that it should be categorized along with other abstract models like Wavetable, Amplitude Modulation (AM), Frequency Modulation (FM), Waveshaping, or Phase Distortion (PD).

10 **Extended Waveform Segment Synthesis (EWSS)**

Waveform segment synthesis techniques play a significant part throughout the history of non-standard synthesis. In this generalized category belong non-standard synthesis approaches like Iannis Xenakis's *Dynamic Stochastic Synthesis*, GENDYN and all its recent variations, G.M. Koenig's *SSP*, Herbert Brün's *SAWDUST*, Arun Chandra's *TrikTraks* and *Wigout*, Gordon Monro's *Fractal Interpolation Waveforms,* or Stelios Manousakis's *Non-standard Sound Synthesis with L-Systems.*

In waveform segment synthesis, minute wave fragments are assembled together to form sound waveforms. These wave fragments are smaller in length than a complete wavecycle. Waveform segment synthesis techniques operate exclusively in the time domain and describe sound by referencing only to amplitude and time values. The time scale of the operations is near the threshold of auditory perception, placing the technique in the territory of microsound. Since this technique generates the waveform with sample per sample operations, it belongs also to the wider category of Direct Synthesis techniques.

This Thesis proposes Extended Waveform Segment Synthesis (EWSS) as a generalized concept that:

- Incorporates all existing waveform segment synthesis techniques into a single model.

- Constitutes the basis for a novel non-standard synthesis model proposed in this Thesis: Dynamic Waveform Segment Synthesis (DWSS).
- Provides a framework for further future research.

10.1 *EWSS: Basic Concepts & Definitions*

In this section we will introduce the basic concepts of EWSS. We will provide definitions for the notions of *segment*, *breakpoint*, *function*, *structure*, *sequence*, and *scheme* that are essential in the synthesis and assemblage of microsonic entities for the computer generation of novel sound objects.

10.1.1 Segment: 1st definition

In EWSS, a waveform is constructed by assembling blocks of amplitude fluctuations with very short durations in the scale of microseconds. They are calculated directly sample per sample. These assembling blocks are called *segments*. The segment is the building unit of this synthesis technique. Each segment can be defined by:

a) a starting breakpoint

b) an ending breakpoint

c) a link (function or shape)

All the synthetic operations in EWSS are actually aimed in the prescription and assemblage of segments.

10.1.2 Breakpoints

Breakpoints are specific junctures in the produced waveform. Every breakpoint is defined by:

a) an amplitude value

b) a time value

The starting and the ending breakpoints of each segment are linked, sample per sample, by a function or a predefined shape.

Since all the segments are joined together one after another, the starting breakpoint of each segment is usually the same to the ending breakpoint of the previous segment. This helps in the continuity of the waveform and avoids unwanted artifacts (distortion, clicks etc). Therefore, we can consider this technique as a succession of breakpoints joined together.

The amplitude and time values of each breakpoint can alternatively take:

- absolute values: each breakpoint is calculated by its amplitude and time values in the wave continuum.
- relative values: each breakpoint is calculated according to its amplitude and time differences from the previous breakpoint.

This Thesis proposes that in EWSS breakpoints can be either derived from external data or generated internally by algorithmic operations.

10.1.2.1 Breakpoint: Data Derivation

Breakpoint values can be derived from external sources. We propose two possible data derivation sources:

- sonification data
- recorded sounds

Although the input values from these sources can be used “as it is”, usually amplitude and time values are selected from longer time frames. These values can be extracted using either:

- a quantization algorithm
- a juncture or transient recognition algorithm.

A more detailed description of this concept is presented in the following section on Segmentation.

10.1.2.2 Breakpoint: Data Generation

Breakpoint data can be generated using a variety of approaches. EWSS incorporates various generative processes that have been already applied in other non-standard synthesis approaches, agglomerates many of them into a single framework and expands them by adding additional data generation models. EWSS proposes breakpoint data generation by using at least the following algorithmic models:

- permutations
- stochastic processes (probabilities, random walks, markov chains, etc)
- chaotic functions (e.g. the logistic map)
- cellular automata
- grammars

An overview of the above models as well as other models used in non-standard approaches is provided in chapter (11) of this Thesis.

10.1.3 Link

Successive breakpoint values are linked together through a function or shape. In other non-standard synthesis implementations (for example SSP, SAWDUSTS or GENDYN) the breakpoints are usually linked together through linear interpolation [Banks *et al* 1979, Brün *et al* 2001, Hoffman 2011]. Nick Collins proposed in his SplineSynth software the use of spline functions [Collins 1999, 2000] while Gordon Monro the use of fractal interpolation [Monro 1995].

This Thesis proposes a variety of link functions or shapes. These can either be generated internally by an algorithm or derived from various external resources. Link shapes can be generated or derived by:

- continuous functions (e.g. linear, spline)
- non-linear functions (e.g. fractal, chaotic)
- recorded sounds
- sonification data
- graphics

We have seen that particular continuous or non-linear functions have already been proposed by other researchers and composer and have been taken part in various historical implementations of non-standard synthesis techniques. One of the contributions of this Thesis is the generalized proposal that any linear or

non-linear function can be used as a segment shape along with shapes derived from other resources like sound segments, sonification data or hand-driven computer graphics.

10.1.3.1 Link: Computer Representation

In most of the non-standard synthesis approaches, each segment is produced algorithmically during synthesis by some kind of linear function, usually by linear interpolation.

We have seen that in EWSS a variety of link function or shapes are applied. For algorithmic integrity and operational efficiency link functions or shapes are stored and recalled in computer lookup-tables that we call *link-tables*. Thus, link functions and shapes, either algorithmic generated or derived from external sources, are stored in computer memory.

Since each segment lasts from microseconds to a few milliseconds, usually it requires a small amount of computer memory. Thus, the size of each link-table does not usually exceed 256 or 512 memory indexes. Simultaneously used link-tables can have arbitrary sizes. For algorithmic simplicity, link-tables are read by normalized indexing, in the range from 0 (table start) to 1 (table end).

Additionally, a requirement for EWSS is that the link shapes are all stored with an ascending value direction: the first index of each link-table takes always the minimum amplitude value while the last index takes always the maximum amplitude value. This is further explained in the section on Segmentation. Amplitude values in each link-table are stored normalized, thus taking values in the range from 0 to 1.

10.1.3.2 Link: Functions

Segments may be defined by algorithmically generated data. The output from simple trigonometric functions or polynomials to various non-linear functions like chaotic functions or fractals can be used to generate a variety of synthetic sound segment shapes. These shapes can range from fragments of familiar and widely used wavecycles (eg. sine-waves or harmonic waveforms) to portions of chaotic orbits of strange attractors.

10.1.3.3 Link: Recorded Sound Segments

Sounds with very short duration or portions of a longer recorded soundfile may be used as segments. These microsonic events preserve the transient characteristics and complex sound morphologies of the recorded sound. The assemblage of sound segments is capable of producing sounds characterized by microtemporal detail in a wide range: from the recreation of recorded sound events (when the segments are played in particular order and adjustment) to the idiomatic creation of environmental sounding abstractions. The concept behind the use and assemblage of segments derived from sound sources is very close to the Brassage technique [Wishart 1994] or the Sample Granulation technique, but transferred and applied to a lower time domain.

10.1.3.4 Link: Sonification Data

Sonification data or a frame of it can define a segment link. Sonification data is mostly derived from natural phenomena but they are not restricted to them. They can be imported in EWSS and stored as a link. For some signals, the data

have to change the time scale to fit the range of the human hearing mechanism and human perception. Therefore, time-stretching and amplitude normalization techniques are required. Moreover, the original signal can be quantized or filtered. An overview on the concept and applications of sonification techniques is presented in section (12.2.4).

10.1.3.5 Link: Graphics

Segment shapes can be also designed graphically. Shapes can be drawn by the hand with an onscreen pencil and displayed on a computer monitor. The application of a tablet input devices is very useful. The use of graphical synthesis allows the composer to design the microstructure of sound events in detail [Roads 2002]. Graphical design of wave segments is a fast way of producing a variety of segment shapes.

10.1.3.6 Segmentation

Externally provided data, either in the form of a sound waveform or in the form of sonification data can be segmented into a list smaller entities. The segmentation process provides three different lists:

- amplitude values
- duration values
- link shapes

In EWSS we propose an automated segmentation algorithm that identifies characteristic amplitude junctions in the provided data. For example, a sound waveform may be segmented at the following points:

- zero crossing
- minima & maxima points between each zero crossing

Additionally, more points can be detected by analyzing amplitude transitions, e.g:

- phase direction change

10.1.4 Segment: 2nd definition

Taken into account from the above discussion that a) a segment is defined by two successive breakpoints that are linked together, b) the starting breakpoint of each segment is usually the same to the end breakpoint of the previous segment and, c) a breakpoint can take relative values to the previous one, we can provide a 2nd definition for the segment. Thus a segment can be also defined by:

a) an amplitude value

b) a duration

c) a link

The amplitude value is actually the value of the ending breakpoint. The duration is the time difference between the starting and the ending

breakpoint. A link is a function or shape that connects the breakpoint ends of the segment.

10.1.5 Hierarchical levels: Segments, Structures, Groups, Sequences & Sound Objects

In EWSS sound construction takes a hierarchical approach: from small waveform fragments towards complete sound objects. Although in EWSS hierarchical construction approach may have an arbitrary number of levels, we propose at least four different levels:

- the segment level
- the structure level
- the group level
- the sequence level
- the sound object level

The segment level is the lowest construction level and the sound object is the highest construction level. Although there may be an arbitrary number of levels in between, we propose at least three intermediate levels: the structure level, the group level and, the sequence level. As we have already seen in chapter (2) all sound construction levels that are below the sound object belong to the territory of micro-level. In the above hierarchical approach, sound objects generated in the context of EWSS may exhibit a vast range of their internal structural complexity. In the next section there will be presented the concepts of *structure*, *group*, *sequence* and *sound object*.

10.1.5.1 Structure

We have seen that segments constitute the building units in EWSS. Segments are assembled together and form larger entities. The smallest sound entity that is formed by a number of segmets is called *a structure.* A structure may be constructed from a single segment up to an arbitrary number of segments predefined by the user. The structure is the minimum synthetic construction in EWSS. A structure is defined by:

- a list of amplitude segment values
- a list of duration segment values
- a list of link shape-tables

The actual form and meaning of the structure is derived from the musical context that EWWS is taking place. For example, the concept of the structure may correspond to the concept of a wavecycle in GENDY, a pulsaret in Pulsar synthesis or a grain in Granular synthesis.

10.1.5.2 Group

In EWSS a structure is usually subjected to algorithmic variations and evolution. The assemblage of successive variations of a structure constructs a larger entity called *a group.* In other words, a group is articulated by a number of [n] successive structures, where each of them is usually an algorithmic variation of the former. A group may be constructed from a single structure up to an arbitrary number of sequential transformations of the initial structure.

A group can be defined by:

- an initial structure
- a length of [n] generated structures
- a transformational algorithm

The degree and the form of the variation between succeeding structures depends on the chosen transformational algorithm. In the simplest case each generated structure is an exact copy. A transformational algorithm may vary in each structure generation any or all of the syntactic elements of a structure:

- the amplitude values of the segments
- the duration values of the segments
- the link shape-tables

A transformational algorithm may be of any complexity and combine one or more simpler algorithmic processes. For example, a transformational algorithm may consist of two nested random walks (similarly to GENDY). Additionally the elastic barriers of each random walk may be modulated by some mathematical function (e.g. a sinewave). Although there may be used a vast number of transformational algorithms, in EWSS we propose the use of at least of the following categories:

- permutation
- stochastic
- chaos

- evolutionary
- fractal
- grammars

10.1.5.3 Sequence

In EWSS a number of groups may be assembled in a higher-level structure called *a sequence.* The complexity of a sequence may vary from a simple occurrence of a single group to complex structures formed by a large number of different groups. A sequence is defined by:

- an initial list of groups
- a length of [n] generated groups
- an ordering algorithm

The ordering algorithm can choose among groups that are provided in an initial list and assemble them in a sequence. A sequence is constructed by a number of [n] successive groups. A sequence may be constructed from a single group up to an arbitrary number of groups predefined in a list by the user.

An ordering algorithm may be of any complexity and combine one or more simpler algorithmic processes. For example, an ordering algorithm may consist of one nested random walk whose step is controlled by a chaotic process while its elastic barriers are controlled by envelope generators. Although there may be used a vast number of ordering algorithms, in EWSS we propose the use of at least of the following categories:

- permutation
- stochastic
- chaos
- grammars

10.1.6 Higher-level Hierarchy

The concept of the hierarchical structure of EWSS can be of an arbitrary depth. There can be as many different levels as the user can define. Every additional structuring level may share similar properties to the concept of the sequence level. Thus, the relationship between groups and sequence can be repeated on different levels. We can easily imagine assembladges of sequences forming a higher-level sound structure and so on.

Morovere, if the hierarchical levels share the same or similar properties, especially the same ordering algorithms, than they can produce complex assembladges with *self-similar or fractal construction*.

10.1.7 EWSS & the Concept of the Sound Object

We have seen that the synthesis structure in EWSS takes the form of hierarchical levels that ranges from small microsonic wave fragments up to the whole construction of a sound object. The total structure provided by the highest hierarchical level should be considered as equal to the internal macro-structure of the generated sound object. For the scope of this Thesis we have presented the hierarchy of four different levels: *segment*, *structure*, *group*, and *sequence*.

Therefore, in the framework of this Thesis, the internal structure of the sound object that is generated by EWSS equals the above-mentioned hierarchy.

11 **Dynamic Waveform Segment Synthesis (DWSS)**

In the previous chapter we have presented Extended Waveform Segment Synthesis (EWSS) in an attempt to (i) incorporate existing waveform segment synthesis techniques into a generalized framework and, (ii) provide a novel paradigm of non-standard synthesis with dynamic algorithmic models.

In this chapter we will propose *Dynamic Waveform Segment Synthesis* (DWSS) as a new non-standard direct synthesis model. DWSS implements the concept of EWSS in a computer music environment. DWSS uses the notions of *segment*, *structure*, *group*, and *sequence* and apply them in a collection of dynamic microsound algorithmic procedures for the generation of novel sound objects.

Although DWSS is a novel approach on non-standard sound synthesis, it takes into account and incorporates various concepts and features from other non-standard approaches like G.M. Koenig's *SSP*, Herbert Brün's *SAWDUST*, Arun Chandra's *TrikTraks* and *Wigout*. However, DWSS should be considered as an offspring of the Stochastic Sound Synthesis and the GENDY systems developed by Iannis Xenakis. Taking into account the relations with the above-mentioned non-standard systems, we will provide direct references to the above systems throughout the description of DWSS for the purposes of comparison and clarity of the concepts.

11.1 ***The computer music programming environment***

DWSS algorithms were developed in the computer music programming environment of Max/MSP. This includes the sound synthesis engine, the algorithmic control structures and the user interface.

Having in mind that this system could be expanded in subsequent implementations, some of its components were intentionally kept simple. This is the case, for instance, of the synthesis engine, which at the time of this research is programmed to operate in non-real-time.

However, all the concepts of EWSS were implemented. Non-standard synthesis with dynamical models is achieved with a group of five separate applications that serve for initial segment list generation, sound segmentation, segment list values transformation, segment list order transformation, structure evolution in groups, group evolution in sequences, synthesis and the relevant user interfaces.

Additionally, the user interface is kept simple in order to serve the merely objectives of this Thesis and it is not designed for any operation of the applications by the public. However, all aspects of software functionalities are taken into concern and are carefully represented in the user interface.

In the following Sections, DWSS's main components are described.

11.2 *System Overview*

In DWSS a sound object is generated algorithmically by the definition, transformation and articulation of a number of short waveform fragments. The program applies the notions of *segment*, *structure*, *group*, and *sequence* that are proposed by EWSS.

The user defines the procedures and provides the control data for the algorithmic generation of sound in six applications that share data, are interconnected and should be considered as operating in parallel.

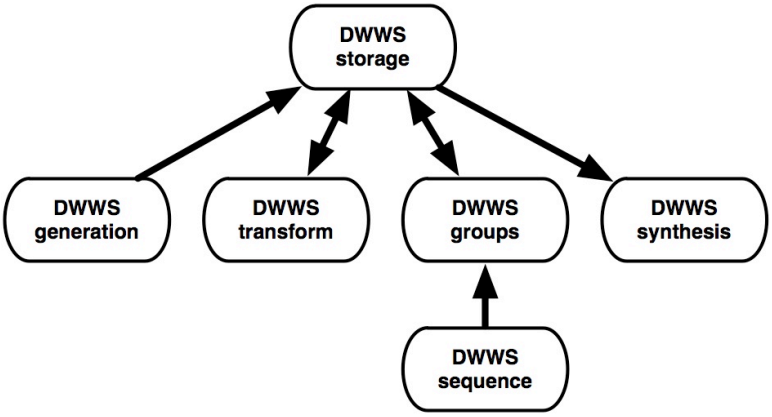

Figure 1: system overview of DWSS

The *"DWSS_storage"* application is responsible for the storage of structures, groups & sequences.

The *"DWSS_generation"* application is responsible for the tasks of a) generation of structures, and b) segmentation of soundfiles into structures.

The *"DWSS_transformation"* application is responsible for the tasks of a) re-ordering of the segments within a structure, and b) transformation of the values of the segments within a structure.

The *"DWSS_group"* application is responsible for the tasks of a) the definition of groups, b) storage of groups, and c) the CA evolution of structures into groups and sequences

The "*DWSS_sequence"* is responsible for the control of the assemblage of groups into structures

The *"DWSS_synthesis"* is responsible for the tasks of a) sound synthesis of sequences, b) playback of synthesized sound, and c) storage of synthesized sound.

11.3 *DWSS:storage*

In DWSS, structure lists are represented as computer tables and thus are accessed and processed through lookup-table techniques. The *"DWSS_storage"* application is responsible for the dynamic memory allocation and the creation or destruction of memory buffers. Each structure list stores segment data in three buffers: segment amplitude values, segment duration values, segment shape number along with data for each segment shape that are stored on separate buffers.

Additionally, the *"DWSS_storage"* application collects and displays information on the current lists:

- the list number
- the number of amplitude values
- the number of duration values
- the number of assigned segments
- the values's mode (whether amplitude and duration values are normalized within the range 0.-1. or have absolute values)
- the length of each segment shape (in samples)

no	amp	dur	shape	mode	Shape lengths														
1	100	100	100	norm	100														
2	8	8	1	norm	100														
3	294	294	294	abs	10	4	3	3	13	17	35	15	9	9	10	10	11	10	10
4	-1	-1	-1	NEW															

Figure 2: list information in *"DWSS_storage"*

11.4 *DWSS:Construction*

In “DWSS_construction” application, each list represents one structure. The *structure* is the smallest sound entity formed in DWSS. A *structure* is assembled together by a group of *segments.* As we have already seen in EWSS in section (14.1.5.1), a structure is defined by a set of values specifying the amplitude, the duration, and the link-shape for each segment.

The first user task in DWSS is the construction of the initial data-lists. These lists will eventually provide the pull for the selection of the appropriate *amplitude* and *duration* values as well as the *shapes* that take part in the construction of waveform *segments* and *structures.* Thus, the user constructs three initial lists according to the requirements of segment definition in sections (14.1.1 and 14.1.4):

- list of amplitude values
- list of duration values
- list of shapes

Lists can be constructed in four possible ways by the operations of: algorithmic *generation*, *soundfile segmentation*, algorithmic *transformation* and group *sequencing*. We will discuss the first two operations in this chapter. List construction through algorithmic transformation is discussed in chapter (11.5), while the operation of group sequencing is discussed in chapters (11.6) and (11.7).

During the initial list construction, the content values of the lists were indexed in parallel. Thus, amplitudes, durations and, link-shapes are already interrelated.

This allows for single indexing of the three lists by a single algorithmic process. Since each structure list consists by a series of segments, the user has to define the total number of them.

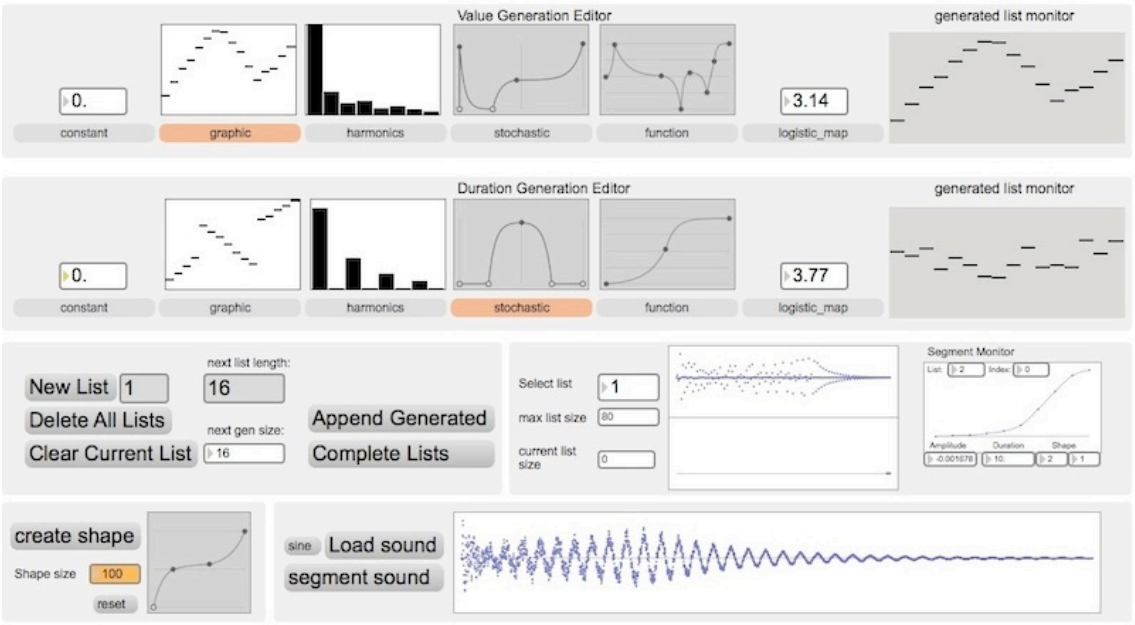

Figure 3: "DWWS_construction" window

11.4.1 Lists & Shapes: algorithmic generation

Lists can be generated algorithmically by applying complex algorithmic functions. The user can choose among the following algorithmic functions:

- constant value. One constant value is applied to all indexes of the list.
- harmonic function. One cycle of composite waveform is generated. The waveform is made up of the weighted sums of 8 simple sinusoids in harmonic relationship.

Figure 4: the harmonic function interface

- stochastic function. The values are generated by a stochastic distribution. The stochastic distribution is defined graphically by the help of a breakpoint function.

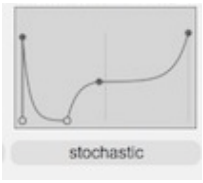

Figure 5: the stochastic distribution interface

- chaotic function (the logistic map). The user chooses the rate of reproduction [*r*].

Figure 6: the logistic map function interface

Although any of the above algorithmic procedures is capable of producing a large variety of numerical lists, DWSS provides the option of chaining together serial generations for even further variety and complexity. There can be as many serial generations as the total length of the list permits.

11.4.2 Lists & Shapes: graphic generation

DWSS provides a user interface for the graphic generation of lists. The system provides the user with the appropriate graphic tools with which the relative lists are filled. Two insertion modes are provided:

- free-hand drawing: the user fills the graphic table by hand movements with the aid of an on-screen pencil tool. The use of a graphic tablet is a useful option.

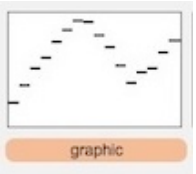

Figure 7: graphic drawing of list values

- breakpoint functions: the user defines specific breakpoints in the table and the computer provides intermediate data through interpolation. The user can select linear interpolation or define any curvature between breakpoints.

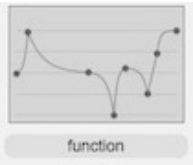

Figure 8: breakpoint function

In the two-dimensional graphic tables, the horizontal axis represents the table index while the vertical axis represents the *amplitude* or *duration value*.

Similarly, the user can define, with the help of breakpoint function, one or more segment *shapes.* To assure signal continuity during segment articulation, the first table index should take the minimum value, while the last table index, the highest. The user defines also the length of the segment.

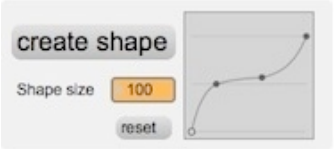

Figure 9: segment shape creation with breakpoints

Since the segment data types (amplitude, duration & shape) are interrelated, the final list should have the same number of amplitudes, durations and shapes. If the number of shapes is less than the amplitude and duration values then the system can complete the list by assign serially the already defined shapes to the unoccupied indexes. This operation is carried by the “complete list” function.

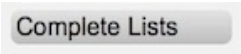

Figure 10: complete list

11.4.3 Lists & Shapes: segmentation

List values and shapes can be also provided from given soundfiles or sonification data. An automated segmentation process derives the values and shapes.

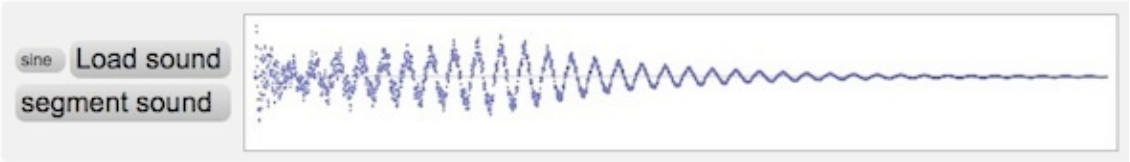

Figure 11: sound segmentation interface

In DWSS we defined a simple segmentation algorithm that identifies zero-points as well as intermediate points with the minimum and maximum values. The three required values that define a *segment* are derived as follows:

- the amplitude value is derived from the value of each identified point
- the duration value is derived by the difference (or in other words the length) in samples between successive identified points

- the shape is derived by the fragment of samples between successive identified points.

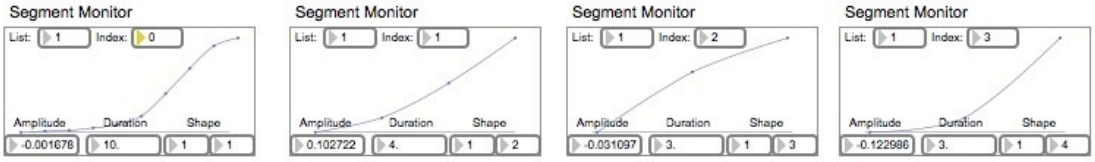

Figure 12: ***four segment shapes***

Since these three data types (amplitude, duration & shape) are interrelated, they are stored in the created lists sequentially and take the same index number.

11.5 *DWSS: algorithmic transformation*

Once the *amplitude*, *duration* and *shape* lists are defined into *structures*, they can be the subjects of further transformation. The *"DWSS_transform"* application, provides two categories of transformations with two distinct operations each:

- index permutation
  - list shaping
  - list random walk
- value transformation
  - add & multiply
  - duration scaling

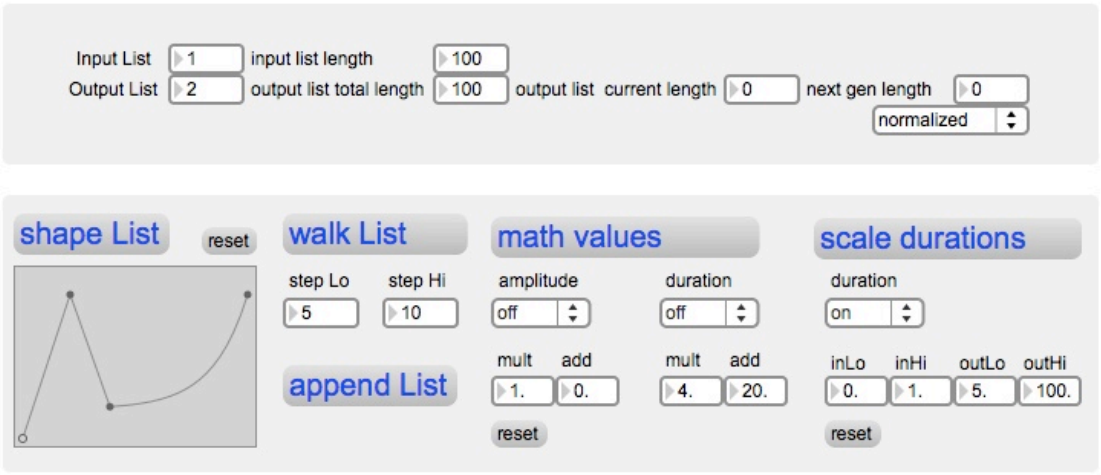

Figure 13: the *"DWSS_trasform"* window

With *index permutation*, the input list indexes are re-ordered with two different operations: a) list shaping and b) list random walk.

In *list shaping* the list is re-ordered with the help of a shaping function that is defined graphically by the user. The horizontal axis represents input list indexes while the vertical axis represents output list indexes. The user defines the shaping function by the help of breakpoint functions.

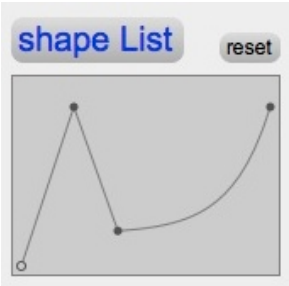

Figure 14: list shaping permutation

In *list random walk,* input list indexes are scanned by the help of a random walk. The user defines the lower and higher length of the step. At each step, the operation selects randomly a) the length of the step, and b) the direction of the

step (forward or reverse). Thus, succeeding chunks of the input list, within predefined limits, are re-ordered and assembled to the output list.

Figure 15: random walk permutation

With *value transformation*, input list values are altered by two different operations: a) add & multiply, and b) duration scaling.

In the *add & multiply* operation, each value of the amplitude or duration segment list is added to or multiplied by a corresponding constant, which is defined by the user.

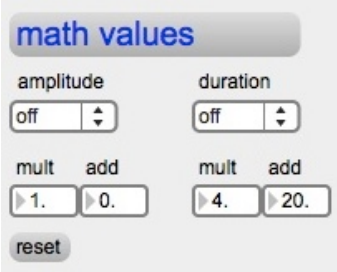

Figure 16: add & multiply transformation

In the *scale durations* operation, each value of the amplitude or duration segment list is scaled within a defined range. Initially, the input list range is automatically set by a function that finds the minimum and maximum value of the list. The user can adjust the ranges of both the input and the output list.

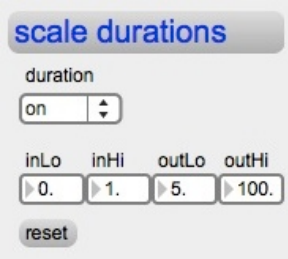

Figure 17: scale duration transformation

Finally, the *append list* operation provides the option of pasting the whole input list at the end of the output list. In this manner two or more list can be combined, making even complex segment structures.

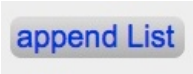

Figure 18: append list operation

11.6 *DWSS: Groups*

A *group* is articulated by an initial list structure and the concatination of a number of its variations. In the *"DWSS_group"* application, a recursive dynamic algorithm generates successive variations of one structure after another. Thus, within a group, every next structure is the evolutionary offspring of the last one. The application of the dynamic algorithm provides continous microsonic variations of various degrees on the generated sound. The initial structure together with the number of generated structure variations defines the length of the group.

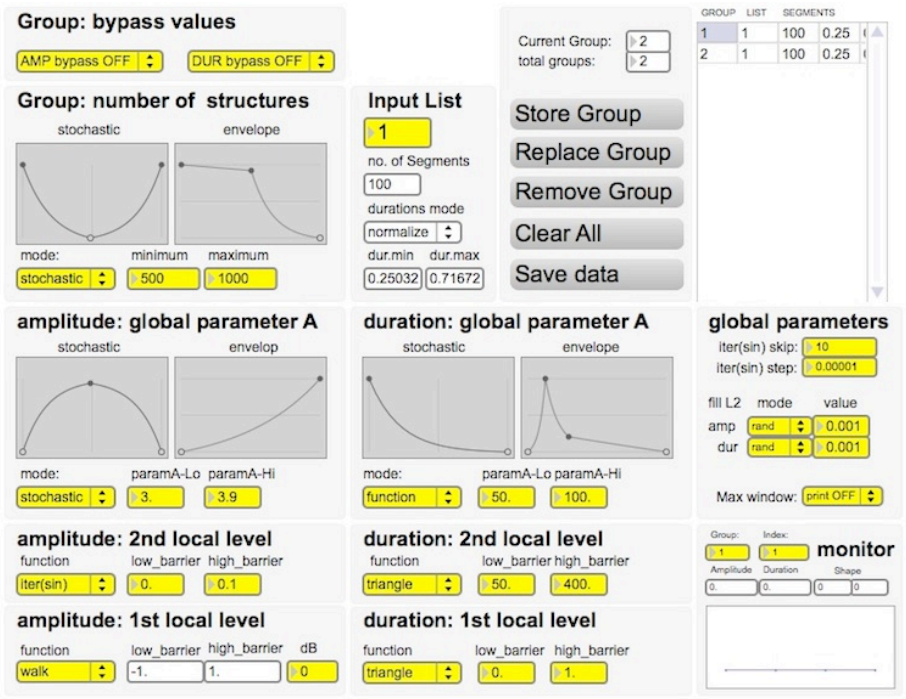

Figure 19: the "DWSS_group" window

11.6.1 Groups: structure evolution with CA

As we have already seen, in EWSS a structure is defined by a set of segment amplitude and duration breakpoints and the link-shapes that joints the breakpoints together. In *"DWSS_group"*, as a group is produced by generating one structure after another, both the amplitude and duration values of the segment breakpoints vary after each new structure generation. In the current implementation of *"DWSS_group"*, the link-shape of each segment remains unaltered.

In *"DWSS_group"*, the user selects the input list structure among the available lists that are provided by the *"DWSS_storage"* application. The input list defines the initial set of segment values to the CA automaton or in other words, the *initial conditions.*

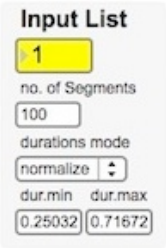

Figure 20: input list

The set of amplitude and duration segment values correspond to individual cells of the CA. Thus, the two CA are *1-dimensional arrays* with a number of cells that is equal to the number of segments that define the input list structure. In other words, *the CA size* is equal to the number of the segments of the input list structure.

Since the generative process of the CA depends on the interaction between neighbor cells, a special attention should be made on the processing of the cells on the boundaries of the 1-dimensional arrays. In *"DWSS_group"*, the *boundary conditions* of the CA are wrapped around, thus forming a torus.

The output of the CA affects directly the computation of the waveform of the sound object therefore it requires the computation of numbers with *floating-point precision*, within the range of [-1, 1]. Actually, in *"DWSS_group"* the CA generates real numbers within the range of [0, 1] that are linearly remapped in the user specified output range. Therefore, the state of each cell has to be continuous with infinite number of states. Thus, the construction engine of *"DWSS_group"* is driven by the special category of *continuous CA* [Wolfram 2002].

There are different *transition rules* for each CA. The transition rules specify the new state of each cell or in other words specify the new breakpoint values of the next generated segment. Thus, the transition rule of the CA is actually responsible for the microsonic variation within a group.

In *"DWSS_group"*, the variation of both the amplitude and duration breakpoint values of each segment are controlled by two independent Cellular Automata (CA) operating in parallel. The *"DWSS_group"* application offers a variety of transition rules, thus is capable of exploring a large number of sonic transformations. Actually, the user selects between two layers of algorithmic functions, one (the upper layer or 2nd local level) providing control parameters to the other (the lower layer or 1st local level). Each function and its input parameters correspond to a separate transition rule.

Figure 21: amplitude Level 1 & 2 functions

The provided algorithmic functions for each layer are:

- cyclic phaser
- cyclic triangle
- cyclic sine
- stochastic random walk
- chaotic logistic map
- chaotic iterative sine

 weighted value from neighbor cells

The values produced by each function are bounded within a user-defined range. These *boundaries* are reflective, thus returning values that exceed the limits back to the nominal levels.

Additionally, *"DWSS_group"* provides automated control to the input parameter (called global parameter A) of the upper level. There are two available modes: a) stochastic distribution, and b) envelope function. Both modes are graphically defined by the user. The parameter A values generated by the selected function are also limited within a user-defined range.

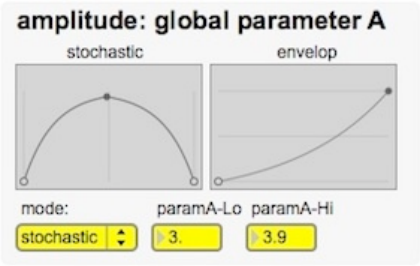

Figure 22: global parameter A

Sometimes the segment values of either the amplitude or the duration structure is sensible to remain unaltered, without any evolution through the CA. In this case *"DWSS_group"* can be set to bypass the amplitude or duration process.

Figure 23: amplitude or duration bypass

The total number of structures generated by *"DWSS_group"* is defined automatically by a specific function. The system provides two modes: a) stochastic distribution, and b) envelope function. Both modes are graphically defined by the user. The selected mode decides automatically for the total number of structures for the current group within a pre-defined range (minimum and maximum number of structures).

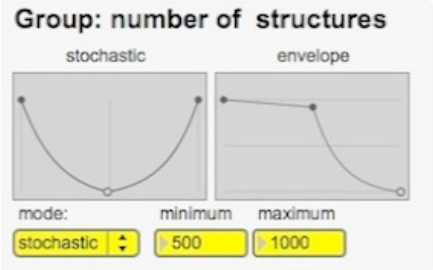

Figure 24: number of structures

Finally, a group is defined by the input list structure, along with all the selected generative functions, the control values, the boundary levels and the total length function. *"DWSS_group"* provides the functionality of storing, editing, recalling and deleting a number of different groups.

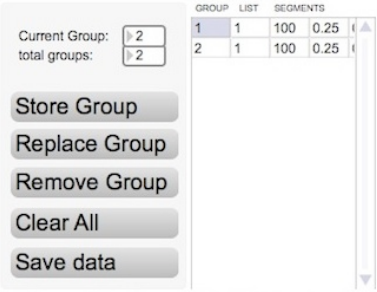

Figure 25: list of groups

Comparing the concept of breakpoint recurrent variation in DWSS with Iannis Xenakis's GENDY system [Xenakis 1992] we can distinguish that DWSS

expands this concept and uses the more generalized notion of CA. In this context, the 2nd order random walks of the GENDY system, that provided the stochastic variations of the breakpoints, is but a subset of the possible transition rules of the CA in DWSS. Moreover, the floating-point precision of DWSS differs from the discrete number of states of the GENDY system, a requirement that highly affects the produced sonorities and their microsonic properties. Peter Hoffman provides a detailed description of the effect of the application of discrete states within GENDY [Hoffman 2009].

11.7 *DWSS: Sequence*

A *sequence* is articulated by the assemblage of groups. A sequence may consist of a single group or by a number different groups. Each group may appear one or more times within a sequence. The *"DWSS_sequence"* application is responsible for the construction of groups into sequences.

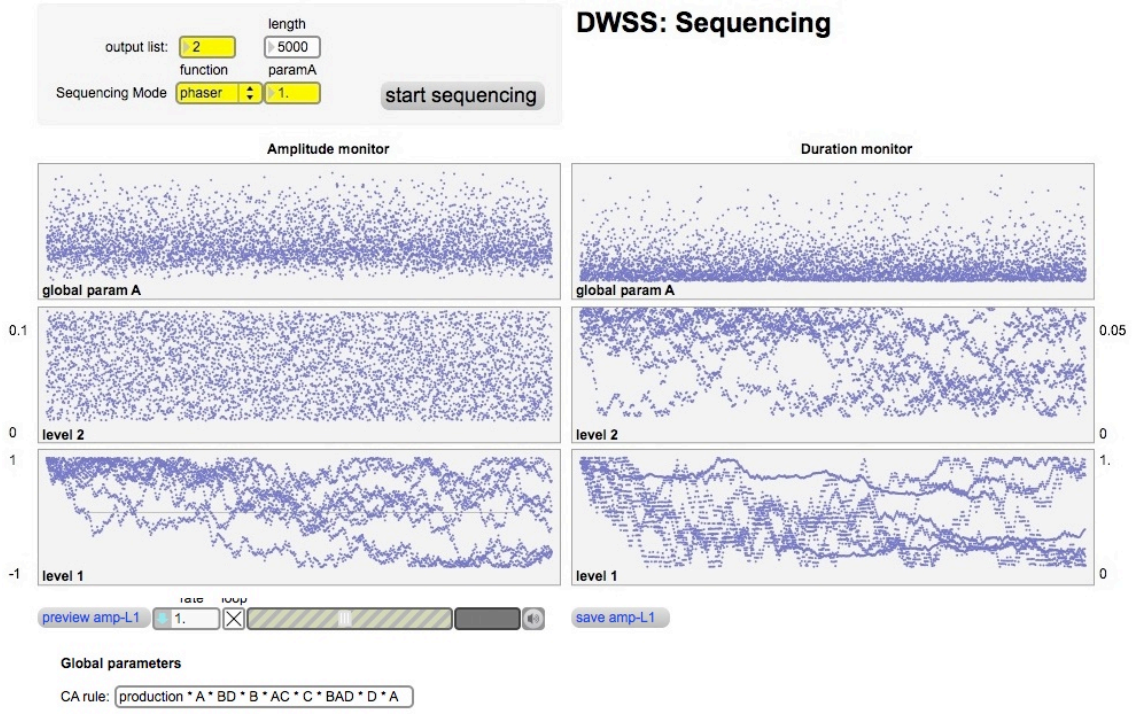

Figure 26: "DWSS_sequence" window

The list-of-groups are defined from the *“DWSS_group”* application. Sequences are generated and assembled by the algorithmic selection of specific groups from the list-of-groups. The list-of-groups is indexed algorithmically by applying complex computer algorithmic procedures. The available functions are:

- oscillating phaser
- oscillating triangle
- stochastic random walk
- chaotic logistic map
- L-system

The selected function is provided with a single input parameter.

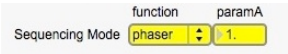

Figure 27: sequencing function

Additionally, in the case of the L-system the user provides also the production rule in the symbolic form of MaxMSP.

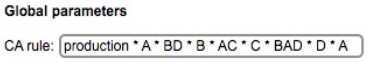

Figure 28: CA production rule

For visualization purposes, the *“DWSS_sequence”* displays the values of global parameter A, Level 2, and Level 1 for both the amplitude and duration of the generated segments.

For sonification purposes, *“DWSS_sequence”* provides playback option for the list of the Level 1 amplitude values. This is especially practical in the special case where all the durations are set to 1 sample. In this case, the list of the

Level 1 amplitude values consist the actual sound object. Level 1 amplitude values can be also saved as a soundfile.

The generated sequence is saved as separate amplitude, duration, and shape lists in *"DWSS_storage".* These lists can be treaded afterwards as any other lists of structures. In this approach sequences can be the input of a new group and thus, we can have sequences of sequences.

For technical reasons, part of the algorithmic processes of assembling groups into sequences where implemented in the *"DWSS_group"* application and controled from the *"DWSS_sequence"* application.

11.8 *DWSS: Synthesis*

The final process of articulating the amplitude, duration, and shape data for each segment, assembling them into a sound waveform and producing the audible *sound object* is undertaken by *"DWSS_synthesis".* The generated waveform can be played back in various speeds for monitoring purposes and better assessment of its sound morphologies. Finally, the generated waveform can be saved as a soundfile.

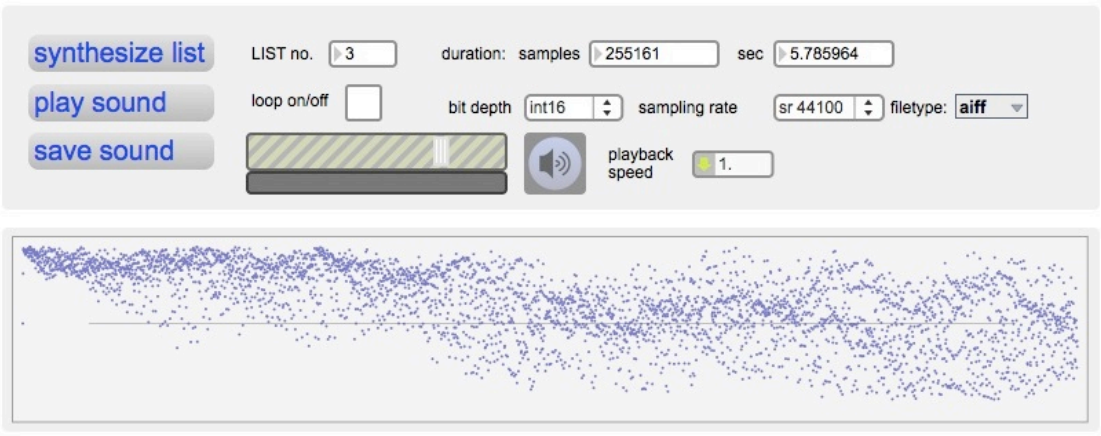

Figure 29: "DWSS_synthesis" window

12 EWSS & DWSS in the Context of Non-Standard Synthesis

The main focus of this Thesis is the proposal of a generalized concept that incorporates all existing waveform segment synthesis techniques into a single model (Extended Waveform Segment Synthesis - EWSS) as well as the realization of a novel non-standard synthesis model (Dynamic Waveform Segment Synthesis – DWSS) which would allow the exploration of various non-standard synthesis ideas derived from that concept.

In this chapter we will try to place EWSS and DWSS in the context of non-standard synthesis. We will attempt to describe the novelty of the concept behind EWSS and the potentiality in synthesizing original sound objects with DWSS. Accordingly, we will first discuss the framework of comparison and evaluation between computer aided algorithmic composition (CAAC) systems in general and non-standard synthesis systems in particular. Consequently, having established the appropriate evaluation framework, we will attempt to place EWSS and DWSS side by side with other non-standard synthesis approaches.

12.1 *On the Comparison of Computer Music Systems*

The utility of computer aided algorithmic composition systems is not easily evaluated. Systems for music generation, or for any generative aesthetic production in general, cannot be assessed merely by its output. The generative results are highly depended on the users, their skill, musicianship and creativity

with the system as well as with their interpretation and utilization of the music or sound results. In this aspect, computer music systems are like any other creative tool or instrument, such as a paintbrush or a violin. The creative results of a tool cannot be entirely measured by analyzing one or more produced instances. Regarding these observations, but talking on hardware rather than software, Gustav Ciamaga suggests that [Ciamaga 1975, pp.117]:

> *"no machine or system has been proven superior to any other despite the claims and counterclaims of designers, manufacturers, or composers… ideally, the composer chooses among the available electronic music systems for their creative potential and not because of any claims of efficiency."*

The efficiency of an aesthetic production system cannot be evaluated by a precise standard of measurement. B.W.Pennycook, surveying computer music systems, suggests that in contrast to other ordinary systems, "in which measures of productivity can be gathered empirically, in most musical settings productivity and aesthetic value become hopelessly confused" [Pennycook 1985]. Music or sound creation computer systems are even more difficult to compare and evaluate. Simon Holland suggests that "in open-ended domains such as music composition, there are in general no clear goals, no criteria for testing correct answers, and no comprehensive set of well-defined methods" [Holland 2000]. Computer music systems should be rather evaluated toward determining limits and potentiality.

If the condition of evaluating computer music systems in general is questionable, the situation is more controversial for non-standard synthesis systems. We have already seen in chapters (5, 6 and, 7) that the non-standard approach belongs to the heretical and explicit culture of computer music. In this

cultural area the objective is the design of algorithmic processes as part of very personalized tools with which the compositional or sound design process is carried out. In non-standard synthesis the composer does not rely on culturally established materials and models, or on any historical experience and practice, but rather defines and implements his own rules and procedures for the generation of the music material. Non-standard synthesis is radically creative instead of reproductive. Although non-standard approaches provided novel models of sound, they are not meant to make available a generalized algorithmic music framework for wide public use but rather they were initially designed to serve particular musical needs of individual composers. This inherent trend towards constant renewal, through the heretical use of technology, of both the music language and material, makes non-standard systems even more arguable to compare.

Moreover, other more practical issues contribute to the difficulties of comparing between non-standard synthesis systems. Since most of the systems are designed for specific compositional objectives by particular composers, they are not publicly available or the access to them is provided only in particular institutions around the world. Other systems are hardware dependant or the target platform is obsolete, as is the case of the hystorical systems. Additionally, for some of the systems the source code does not exist or they are only superficially documented.

However, as we have already seen in chapter (10), non-standard synthesis, as part of computer aided algorithmic composition, can make particular use of methodologies of cognitive musicology. As Michael Hamman suggests,

algorithmic music systems allow the critical assessment of the compositional process itself [Hamman 2002, pp.93]:

> *"The computer... enabled the composer to critically examine and assess the musical result, the means by which the result came about, and how the two are conceptually and generatively related"*

In algorithmic composition, rules of the composition or of sound design are already "there", as part of the creative process. The identity of the generated sound object is formalized in the algorithmic generation process. In this respect, the formalization of the algorithmic process may be considered as objectified theory. This objectification makes possible the comparison, if not of the complete musical systems, at least some integral parts of them.

12.2 *DWSS & the Other Non-Standard Systems*

A non-standard synthesis system, along with any computer aided algorithmic composition system, may be analyzed by dividing it into components. Generally, we can divide the components of a computer music system into three groups: models of sound/music material, models of sound/music procedures and, the large-scale architecture. This division is in reference to the classic division of software into data structures, algorithms and, system configuration [Winograd 1979].

As we have seen, concerning the models of the sound material, there are two general groups of non-standard synthesis systems. First, systems where the generated sound object is represented as a construction of individual samples. Second, systems where the generated sound object is represented as a

construction of sound segments. DWSS is enlisted in this latter category. However, since a segment may take the minimal value of one sound sample, this categorization is obscure.

The substantial discretion between non-standard systems lies on the utilized algorithmic models. As we have already see in chapter (6-7), in non-standard synthesis, compositional aesthetics and formalizations are applied to the direct construction of the sound waveform itself. This is a bottom-up approach that is based on computer instructions. The applied algorithmic procedures and models actually distinguishes the particular non-standard systems and are the fundamental base for the potential sound-object generation.

By reviewing non-standard synthesis systems we arrived at specific algorithmic models that provided distinctive functionality and were responsible for the generation of characteristic sonological features of the sound objects. These algorithmic models are: list generation, list permutation, tendency mask, trigonometric functions, stochastic variation, chaotic variation, cellular automata and grammars. These algorithmic models, their significant features and, how they contributed to particular sound models and synthesis approaches were presented in chapter 11. We will compare the implementation of these algorithmic models to various non-standard synthesis systems along with DWSS. These systems are: G.M. Koenig's SSP, Herbert Brun's SHAWDUST, Arun Chandra's TrikTraks and Wigout, Jaques Chareyron's LASy, Iannis Xenakis's GENDY, Stelios Manousakis's Non-standard Sound Synthesis with L-Systems (NSSSLS) and Agostino di Scipio's Iterated Function Synthesis (IFS).

The concept of algorithmic *list generation*, along with algorithmic list permutation was applied in SSP, SAWDUST, TrikTraks and Wigout. Furthermore, SSP introduced the concept of *tendency masks*. TrikTraks and Wigout expanded the concept of SAWDUST by incorporating *trigonometric functions* among others. *Stochastic variation* is an integral aspect of the GENDY system and its successors. *Chaotic variations* are explored in Iterated Function Synthesis (IFS) along with other Agostino di Scipio's implementations. The use of *cellular automata* was first introduced in direct sound synthesis in LASy. Finally, the utilization of *grammars* is extensively explored in resent research in Non-standard Sound Synthesis with L-Systems (NSSSLS). Table (1) provides an overview of the utilization of various algorithmic models in different non-standard synthesis system. The last column demonstrates the integration of all these models in DWSS.

	SSP	SAWDUST	TrikTraks	Wigout	LASy	GENDY	NSSSLS	IFS	DWSS
list generation	X	X	X	X					X
list permutation	X	X	X	X					X
tendency mask	X								X
trigonometric functions			X	X					X
stochastic variation						X			X
chaotic variation								X	X
cellular automata					X				X
grammars							X		X
graphics									X

table 1. Comparative Table of Non-Standard Synthesis Systems

However, all these algorithmic models are not isolated features in DWSS that are responsible for the generation of particular categories of sound morphologies. DWSS's architecture provides complex configurations and combination of these models, thus is capable of generating a wider range of sound objects than the stand-alone operation of a single algorithmic model.

Finally, system architecture, the third component of a computer music system, defines how its components interact and are displayed to the user at the highest level. One aspect of this level is the choice of the user interface, graphical or otherwise. Although the user interface is an important aspect of the usage of a computer music system, our research rather focuses on concepts and procedures. The examination and discretion on the basis of system architecture in general and the implication of the user interface in particular is beyond the scope of this Thesis.

13 **DWSS: Case Studies**

Dynamic Waveform Segment Synthesis is capable of generating sound objects that feature a broad range of sonic morphologies. Since in DWSS, sound construction operates in hierarchies, from segment units towards the complete sound object, any differentiation on any level is capable of generating diverse results. Thus, synthesis approaches may focus more or less on initial segment lists, structures, groups or sequences. Any differentiation on any hierarchical level opens a range of possibilities in sound construction.

As we have already seen in the previous chapter, the morphologies of sounds generated by DWSS depend on the following factors:

- initial list(s) properties
- group definition
- sequence definition

In the simplest case, where the sequence consist of only one group, there are already a significant number of factors: (2) parameter A modes [stochastic, envelope] and (7) possible function for Level 2 and Level 1 operations [phasor, triangle, sine, walk, logistic map, iter(sin) and neighbor rule]. That means that there are 98 combinations (2 x 7 x 7), separately, for the amplitude and duration definition of segments. That makes in total 9604 possible combinations (98 x 98) of functions that could define a Group.

To the above complexity one has to consider the number of values for each function as well as the variety of the generated behavior. That makes a vast algorithmic parameter space that requires a systematic research that goes

beyond the scope of this Thesis and it is questionable that can fully explore and categorize all the morphological properties of the possible generated sound objects. However, in order to investigate the basic functionality of DWSS and to present some of its generative properties we conducted a number of basic experiments within the framework of eight (8) case studies.

The eight (8) case studies cover two categories of sound generation approaches. On one hand, we will provide examples that indicate how DWSS is capable of generating sound objects that are characteristic of other approaches. Thus, we will illustrate how DWSS functions as a generalized framework that incorporates basic features of other microsound synthesis techniques. In this category belong the case studies:

1) Soundfile Segmentation and Resynthesis
2) Dynamic Stochastic Synthesis
3) Iterated Nonlinear Functions
4) Oscillating functions & Tendency Masks
5) Sound Synthesis with Graphics & Grammars

One the other hand, we will provide examples disclosing the novelty of DWSS and how it can generate sound objects featuring morphologies that belong to the heretical currents of contemporary computer music creativity. In this category belong the case studies:

6) Dynamic Sound Synthesis
7) Cellular Automata Sound Synthesis
8) Dynamic Microrythmic Morphologies

However, as has already mentioned above, since DWSS belongs to the non-standard synthesis approaches, the provided case studies are but a subset of the potential sound morphologies. Non-standard synthesis requires a lot of laborious experimentation but recompense with the discovery of new sonic territories.

13.1 *Case Study 1: soundfile segmentation and resynthesis*

In the case study described in this section, the aim was to segment and resynthesize an existing soundfile with DWSS. Although DWSS, as any non-standard synthesis technique, belongs to the *explicit culture* of computer music, which is not interested in the emulation or re-creation of existing sound objects, this basic experiment had twofold intention:

- to examine the segmentation and recosnstruction functions of the system.
- to validate that the system can provide the composer with the morphology of an existing sound object as the starting point for further sound mangling and experimentation

For the purposes of this study, we recorded directly into the computer, with the aid of a quality microphone and audio interface, the sound of a small bell. The soundfile was first edited in a sound editor so any silence was removed from the beginning and the end of the recording, resulting in duration of 328 msec.

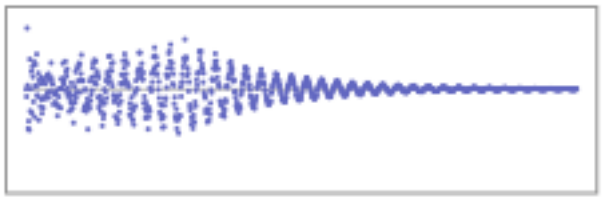

Figure 30: bell sound to segment

With the help of the segmentation option of the "*DWSS_generation*" application, the soundfile was segmented into 294 segments. The amplitude, duration and, shape data of the segments was stored sequentially into the appropriate lists.

For the purposes of this study, the target sound object is the exact resynthesis of the initial soundfile. Since the segment data was stored sequentially, there was no need for any algorithmic transformation of the lists. Thus, the soundfile could be reconstructed with the sequential assemblage of the initial segments.

This operation could be easily performed with the help of the "*DWSS_synthesis*" application. The stored segments were re-assembled with the help of a single function: a linear reading of all the contents of the lists and without any other proccess.

In this perspective, the target sound file was represented as only one structure. What is interesting is that this case study provides also an example of how the final sound object is constructed with operations carried in only one level, the structure level. Therefore, there was no need for further operations in the group and the sequence levels: there was only one group that consist of a single structure without any internal operation of the Cellular Automaton mechanism as well as a sequence that consist of that single group.

The overview of the hierarchical structure of the sound object within DWSS is:

- one long structure representing the initial soundfile
- one group without any internal CA evolution
- one sequence with one group occurrence

Comparing the initial soundfile sample-by-sample to the resynthesized soundfile completed the aim of this case study. The comparison was carried with the aid of a specially programmed algorithm. The comparison resulted in two identical files, thus the resynthesized soundfile was an exact recreation of the initial sound.

13.2 *Case Study 2: Dynamic Stochastic Synthesis*

It is already said that DWSS should be considered as an offspring of the Stochastic Sound Synthesis and the GENDY systems developed by Iannis Xenakis. In this case study, the aim was to emulate with DWSS the sound engine of the GENDY system. Dynamic Stochastic Synthesis and GENDY was introduced and discussed in section (13.1).

The first step was to create the initial segment lists with the help of the "*DWSS_generation*" application. We chose to construct an evolving structure that is assembled by 8 segments, therefore we defined the amplitude, duration and, shape lists with the corresponding length. GENDY generates sounds that always start from silence or as Peter Hoffmann poetically describes it, music out of nothing [Hoffmann 2009]. Thus, we chose to set all indexes of both the amplitude and duration lists with zeroes. This selection was carried by choosing the *constant value* option in the *algorithmic generation* of the lists. Moreover, the GENDY system uses linear interpolation for the linking of the breakpoints.

Therefore, we created a single linear shape of 100 samples duration using the *algorithmic generation* and *straight-line* options.

The second step was to define the group and to emulate the stochastic displacement of the breakpoints. For this purpose we used the “DWSS_group” application. The emulation of GENDY’s sound engine would be carried by the Cellular Automaton mechanism of DWSS: within a group, the variation of the amplitude and duration breakpoint values of the structure would be controlled by two independent CA operating in parallel. For this purpose we defined the creation of a group with the following settings: We chose the same transition rule for both CA. This rule consists of two random walks for each algorithmic layer. The output of the higher random walk was set to control the step of the lower. Although the ordinary operation of CA take into account the values of the neighbor cells, for the purposes of this case study we didn’t used the weight of the neighbor cells and took into account only the stochastic operations within the internal transition rule.

Since the internal structure evolution within a group is sufficient in emulating the stochastic sound generation of the GENDY engine, the final sound object hierarchy requires only one group with one occurrence within the sequence.

For the generated sequence of the segments we used one group with the following values:

- group length: 5000 structures
- amplitude global parameter A: stochastic mode with range 0.-0.1
- amplitude Level 2: walk function with range 0.-0.1
- amplitude Level 1: walk function with range -1. – 1.

- duration global parameter A: stochastic mode with range 0.-0.01
- duration Level 2: walk function with range 0.-0.05
- duration Level 1: walk function with range 0.-1.

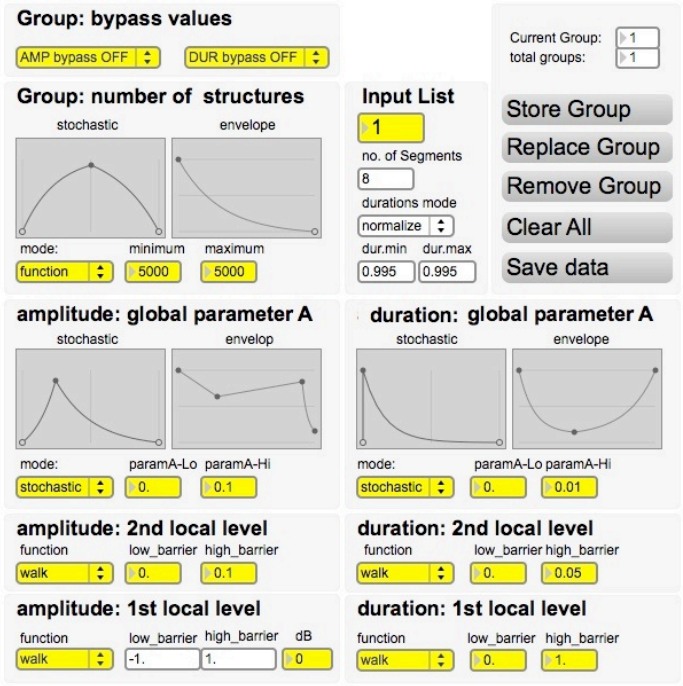

Figure 31: case study 2 group window

Since the durations of generated segments were normalized (0.-1), we further used the "*DWSS_transformation"* application and scaled the durations within the range of 10 – 100 samples.

The overview of the hierarchical structure of the sound object within DWSS is:

- one small structure representing GENDY's wavecycle
- one group with the CA representing GENDY's stochastic evolution
- one sequence with one group occurrence

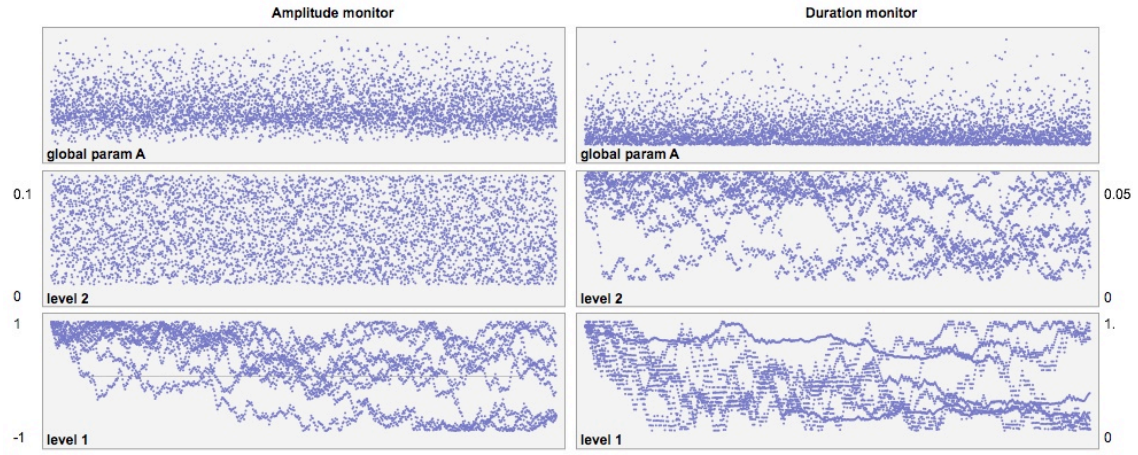

Figure 32: case study 2 sequence window

The requirements for the exploration of GENDY's sound morphologies was the experimentation with:

- different number of segments for the initial wavecycle
- different stochastic distributions for each random walk
- different elastic barriers for each random walk

With the particular hierarchical structure and by taking into account the requirements for GENDY's sound exploration described above, we were able to experiment and synthesize various Dynamic Sound Synthesis sound morphologies.

13.3 *Case Study 3: Iterated Nonlinear Functions*

In this case study, the aim was to emulate with DWSS the Iterated Nonlinear Functions (IFS) approach of Agostino Di Scipio as well as to create a variety of associated sound objects.

IFS utilize an iterative function that operates in the sample level and generates sound objects that exhibit chaotic morphologies. IFS require the generation of a stream of individual values that represent the stream of audio samples. In this case we need to unify the basic construction unit between IFS and DWSS. The construction unit in IFS is the sample while the construction unit in DWSS is the segment. Thus we need to prepare DWSS so the length of the segment equals to one sample. The consequence of this assumption is that only the amplitude parameter of the segment is to be taken into account while both the duration and the shape are actually indifferent.

Accordingly, with the use of the *"DWSS_generation"* application, the initial lists are constructed with only one element. Similarly, only one structure is constructed and the structure consists of only one segment.

The iterative process can be easily implemented in DWSS with the help of the Cellular automaton with the "DWSS_group" application. In this case study, the iterative function, which is actually the *sine map* function, is set as the *transition rule* for the CA. Actually, only the lower of the two layers of the provided algorithmic processes is required. Moreover, since the *sine map iterations* utilize only internal algorithmic processes within the *transition rule*, the evolution of each CA cell is independent of the neighbor cells. Thus, the weight of the neighbor cells was set to zero.

Since the internal structure evolution within a group is sufficient in emulating the chaotic sound generation of IFS, the final sound object hierarchy requires only one group with one occurrence within the sequence generated by the "DWSS_sequence" application.

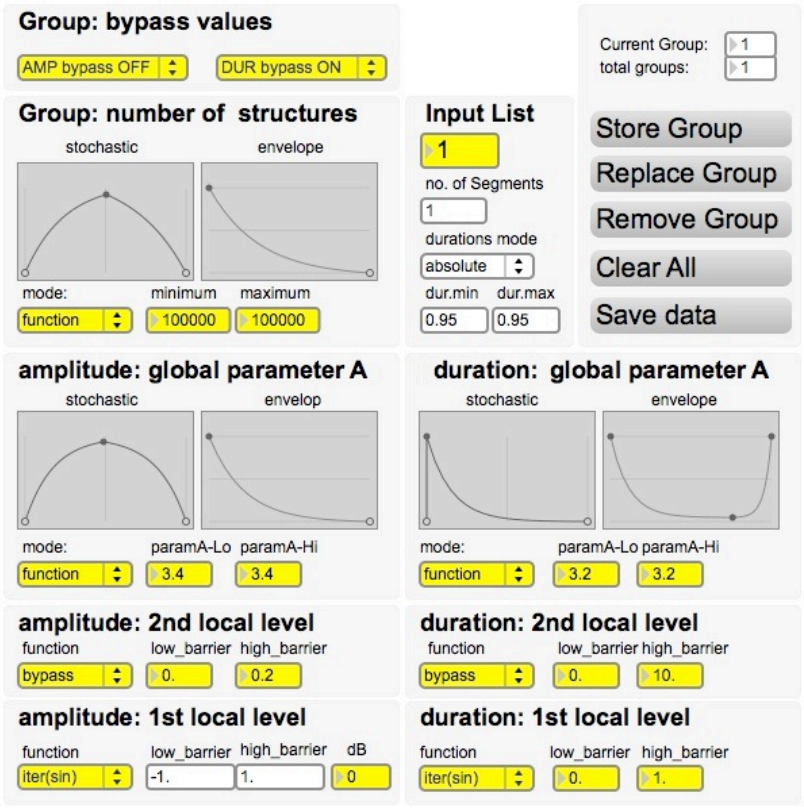

Figure 33: case study 3 group window

For the generated sequence of the segments we used one group with the following values:

- group length: 100.000 structures
- amplitude global parameter A: function mode with range 3.4 – 3.4
- amplitude Level 2: bypass
- amplitude Level 1: walk function with range -1. – 1.
- duration global parameter A: stochastic mode with range 3.2 – 3.2
- duration Level 2: bypass
- duration Level 1: walk function with range 0.-1.

Since the durations of generated segments were normalized (0.-1), we further used the "*DWSS_transformation"* application and scaled the durations within the range of 10 – 100 samples.

The overview of the hierarchical structure of the sound object within DWSS is:

- one structure with only one segment
- the segment has minimal duration equal to one sample
- one group with the CA representing IFS chaotic evolution
- one sequence with one group occurrence

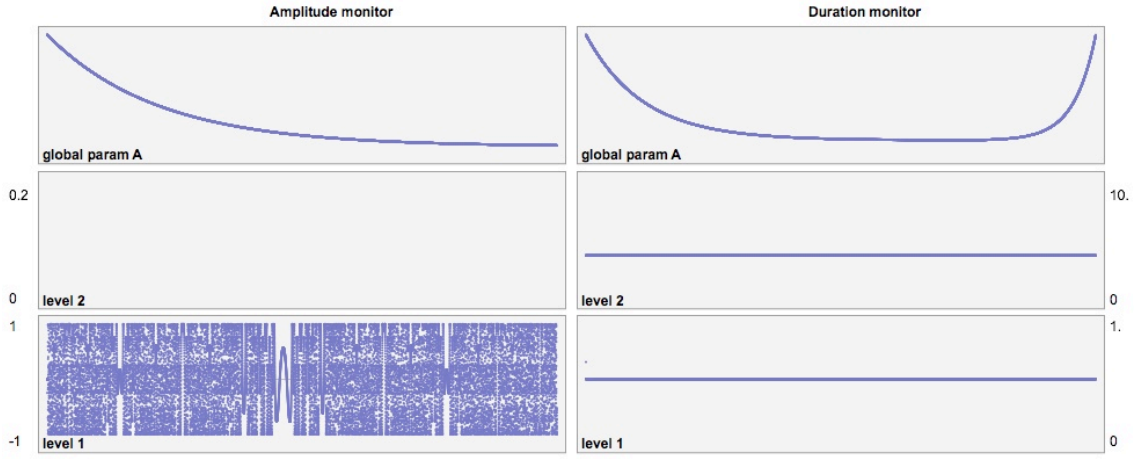

Figure 34: case study 3 sequence window

In general, the requirements for the exploration of IFS sound morphologies is the experimentation with:

- different initial value [x] for the chaotic function
- different scaling factor [r] for the chaotic function

With the particular hierarchical structure and by taking into account the requirements for IFS sound exploration described above, we were able to experiment and synthesize various chaotic sound morphologies.

13.4 *Case Study 4: Oscillating functions & Tendency Masks*

In this case study, the aim was to emulate with DWSS some special and distinctive features of the SAWDUST and SSP systems. In particular we wanted to emulate the oscillating features of the SAWDUST system in parallel with the tendency mask features of the SSP systems.

The oscillating features are able to generate sound objects exhibiting modulating morphologies ranging from simple to complex tremolo and vibrato effects, up to Amplitude Modulation (AM) and Frequency Modulation (FM) sonorities. Additionally, the tendency mask is an expressive function that provides dynamic minimum and maximum value control over other functions, reducing the output of the latter within a specific range and thus providing a powerful control over the morphological evolution of the generated sound object over time.

For the purposes of this study we decided to construct sound objects that rely on long modulating structures.

Initially, we generated, with the help of “DWSS_generation” application, lists with 7 segments each. Next, we used the “DWSS_group” application for the definition of the modulating and tendency mask operations. The chosen algorithmic structure consisted of two oscillating functions, one modulating the other. A tendency mask further controlled the output of the higher level function. In both the amplitude and duration sections, in Level 1 used a sine function, in Level 2 a triangle function and for the global parameter A used the function mode as tendency mask. This procedure was chosen to generate structures

with 5.000 segments length. Particularly, for the generated sequence of the segments we used one group with the following values:

- group length: 5.000 structures
- amplitude global parameter A: function mode with range 50 - 100
- amplitude Level 2: triangle function with range 10 - 25
- amplitude Level 1: sine function with range -1. – 1.
- duration global parameter A: function mode with range 100 - 400
- duration Level 2: triangle function with range 20 – 50
- duration Level 1: sine function with range 0.-1

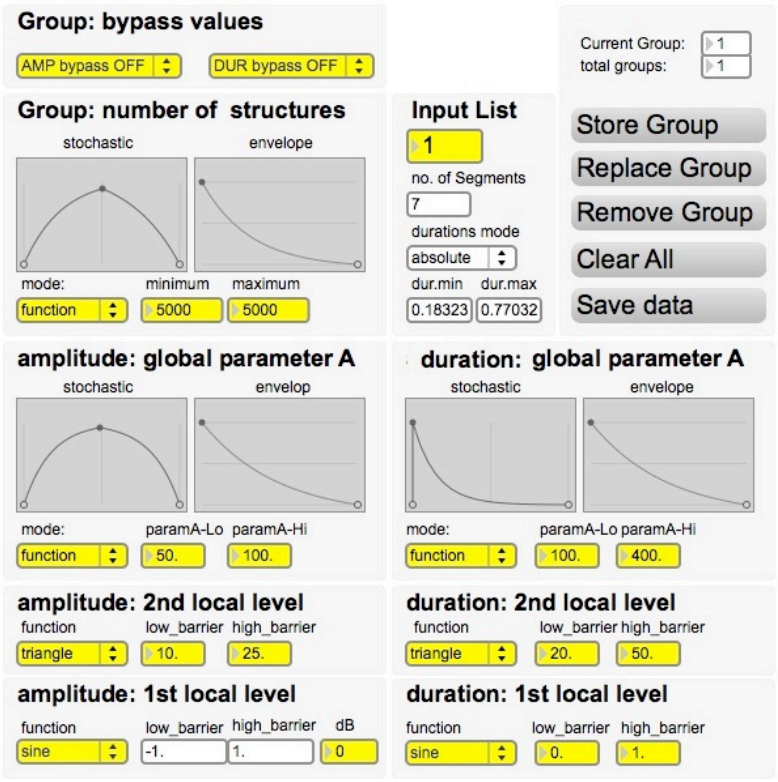

Figure 35: case study 4 group window

For the purposes of this case study, the construction process and the length of the generated structure was sufficient for the obtainment of our aims. Thus, no further processing was necessary for the construction of other structures and

groups. The final sound object hierarchy requires only one group with one occurrence within the sequence.

Since the durations of generated segments were normalized (0.-1), we further used the "*DWSS_transformation"* application and scaled the durations within the range of 10 – 100 samples.

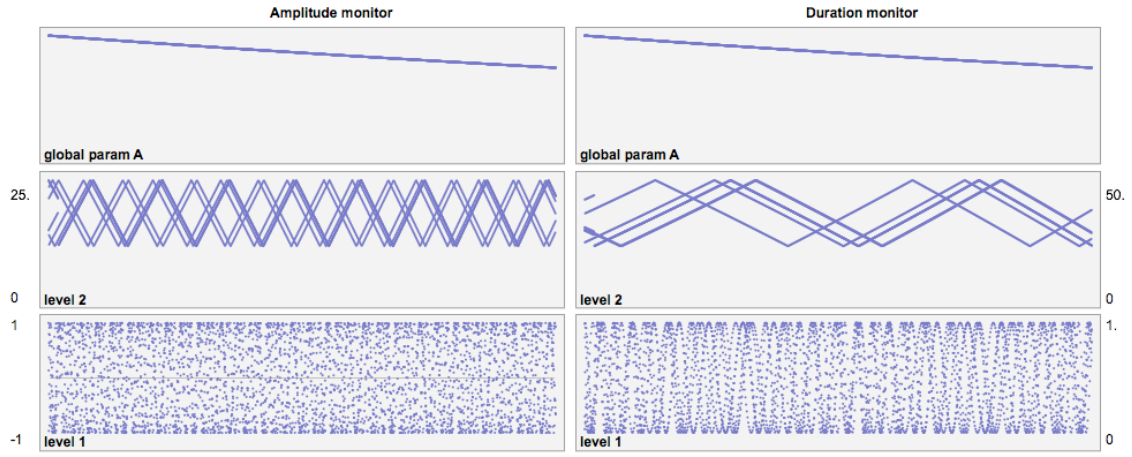

Figure 36: case study 4 group window

The overview of the hierarchical structure of the sound object within DWSS is:

- one structure
- one group with CA evolution through double cyclic modulation and tendancy masks
- one sequence with one group occurrence

The requirements for the exploration of modulating sound morphologies controlled by tendency masks was the experimentation with:

- the length of the oscillating functions
- the degree of modulation of one function with another
- the control of the modulation with tendency masks

With the particular hierarchical structure and by taking into account the requirements for oscillating sound exploration described above, we were able to experiment and synthesize various modulating sound morphologies ranging from simple tremolo or vibrato up to complex Amplitude (AM) or Frequency Modulations (FM). We were also able to control the various levels and degrees of modulations with the help of the tendency masks.

13.5 *Case Study 5: Sound Synthesis with graphics & grammars*

In this case study, the aim was to generate with DWSS novel sound objects by using a formal grammar and a set of formation rules. The concept of sound synthesis by rules has been discussed in the algorithmic models of grammars (section 11.5), Lindenmayer Systems (section 11.6) and, fractals (section 11.4.2).

First, we created with the help of the “DWSS_generation” application two different sets (amplitude, duration, link) of lists. These lists were decided to have very short length, each consisting of 8 values. For the construction of each list, we used the graphic generation option. For both lists were used one linear link-segment.

Next we defined with the help of the “DWSS_group” application two different groups. For both groups we applied the concept of Stochastic Waveform Synthesis where one random walk controls another. For the first group we used the following settings:

- group length mode: function with range: 5 – 10 structures
- amplitude global parameter A: function mode with range 0.01 – 0.4

 amplitude Level 2: walk function with range 0. – 0.1

 amplitude Level 1: walk function with range -1. – 1.

 duration global parameter A: function mode with range 0. – 5.

 duration Level 2: walk function with range 2 – 10

 duration Level 1: walk function with range 5-40

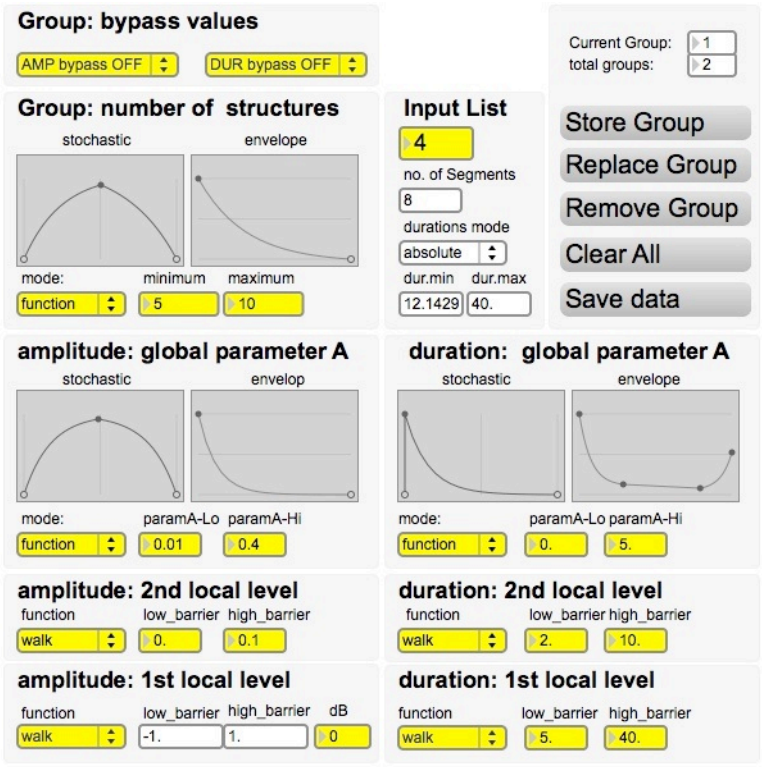

Figure 37: case study 5 group 1 window

For the second group we used the following settings:

 group length mode: function with range: 5 – 10 structures

 amplitude global parameter A: function mode with range 0. – 0.1

 amplitude Level 2: walk function with range 0. – 0.2

 amplitude Level 1: walk function with range -1. – 1.

 duration global parameter A: function mode with range 2. – 10.

 duration Level 2: walk function with range 0. – 10.

 duration Level 1: walk function with range 50 - 100

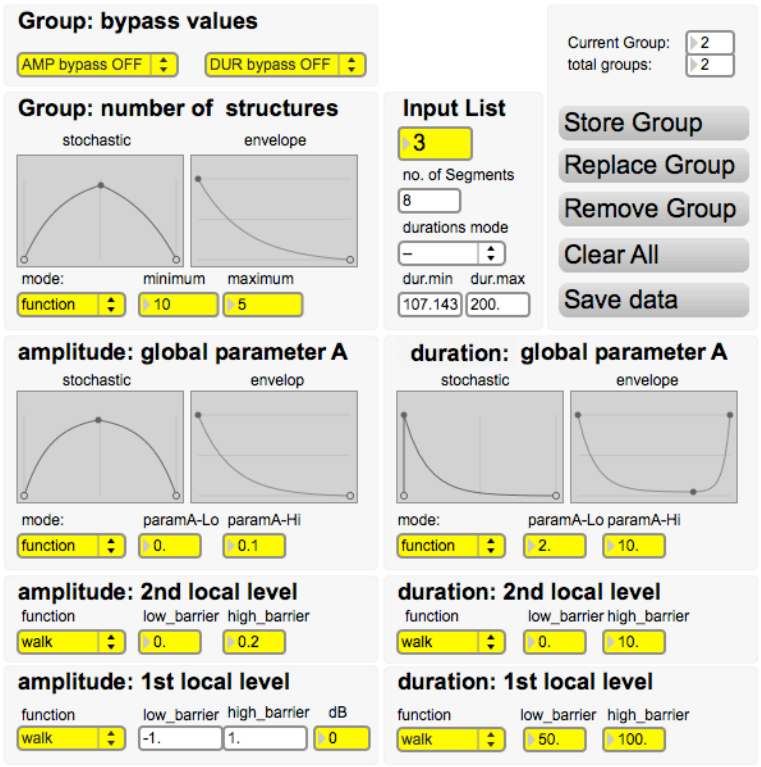

Figure 38: case study 5 group 2 window

next, within the “DWSS_sequence” application we activated the grammars option where we defined the alphabet (A,B) by selecting the two already generated set of list. We provided two simple production rules: (A→AB) and (B→A). These production rules are identical to Lindenmayer's original L-system for modelling the growth of algae. The initial string or the axiom of the grammar, were set to (A). Finally we decided to generate a sequence of 5.000 seqments.

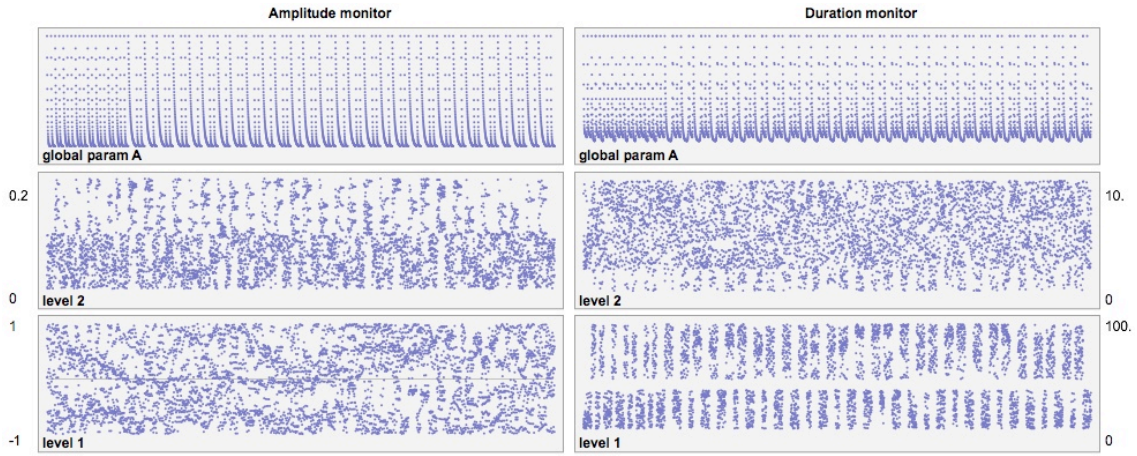

Figure 39: case study 5 sequence window

Since the scope of this case study was focused on the exploration and demonstration of the sound generation function by the application of a simple grammar, we decided that the formatted list was sufficient. Thus, no further processing was necessary for the construction of other structures and groups. The final sound object hierarchy requires only one group with one occurrence within the sequence.

The overview of the hierarchical structure of the generated sound object within DWSS is:

- one very long structure generated by grammars
- one group without any internal CA evolution
- one sequence with one group occurrence

The requirements for the exploration of sound morphologies generated by grammars was the experimentation with:

- the construction of initial list that served as the alphabet of the grammar
- the set of production rules
- the initial axiom

13.6 *Case Study 6: Dynamic Sound Synthesis*

We have already seen in case study 2 that DWSS is capable in providing the framework for dynamic stochastic synthesis. In this case study we wanted to explore some novel possibilities of DWSS in dynamic sound synthesis generation. We used the hierarchical sound object definition of case study 2 as

the basis for further exploration. Actually, we kept all the sound object definition intact. Therefore, we retained the following conditions:

- one small structure representing the evolving wavecycle
- one group with the CA representing the dynamic sound engine
- one sequence with only one group occurrence

The aim of this case study was to explore different types of transition rules that are responsible for the displacement of the waveform breakpoints through stochastic displacement on the lower level. By the term type we refer to the algorithmic formalism that take the form of two layers of processes, the upper layer controlling the lower. As we have already seen, there are four different categories processes available for each layer: cyclic, stochastic, chaotic and classic CA. Since in the cyclic category DWSS provides three different functions (phaser, triangle and sine) we chose to experiment with only one of them, the triangle function. Thus, we have five different possible combinations between level 2 and level 1 functions, that form five different sub-cases:

- triangle → random walk
- random walk → random walk
- logistic map → random walk
- iterative sine → random walk
- CA neighbor rule → random walk

The first sub-case utilizes a random walk whose step or elastic barriers are controlled by a *trigonometric function*. This combination generates oscillating morphologies that range between a static sound (when the trigonometric

function outputs zero) and full stochastic behavior (when the trigonometric function outputs the maximum value). The global parameter A was set to function mode that controlled the length of the triangle function. The settings used within the *"DWSS_group*" application for this case study was:

- group length: 8.000 structures
- amplitude global parameter A: function mode with range 100. – 300.
- amplitude Level 2: triangle function with range 0. – 0.15
- amplitude Level 1: walk function with range -1. – 1.
- duration global parameter A: function mode with range 100. – 500.
- duration Level 2: triangle function with range 0. – 4.
- duration Level 1: walk function with range 2. – 50.

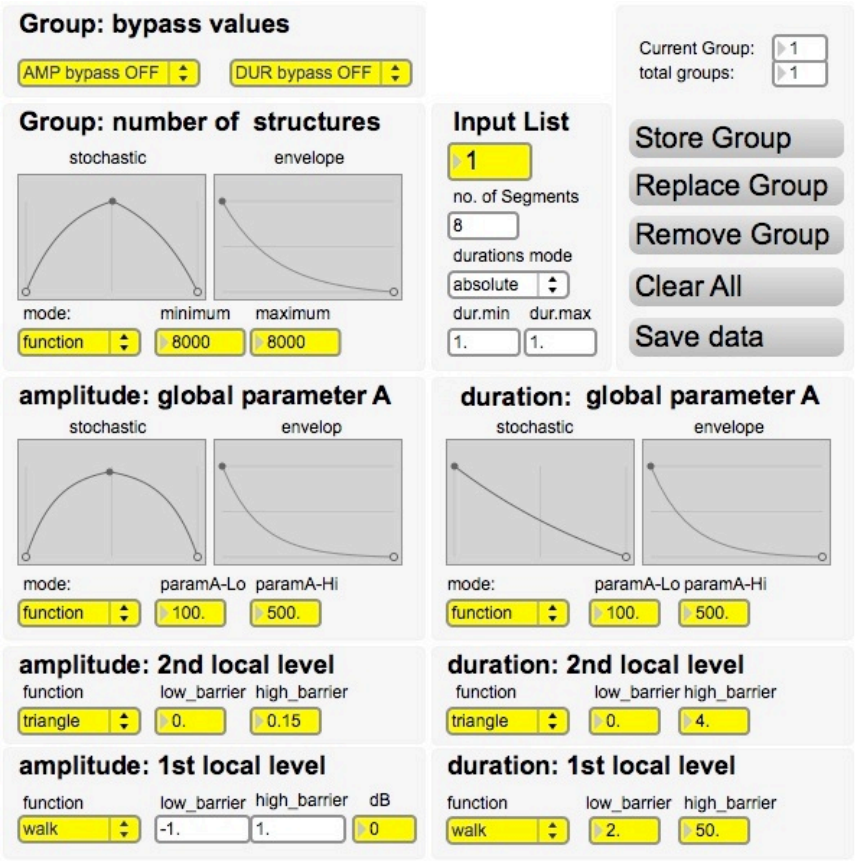

Figure 40: case study 6.1 group window

The behavior of the triangle function controlling the walk function can be observed graphically from the corresponding monitor windows of the "*DWSS_sequence*" application.

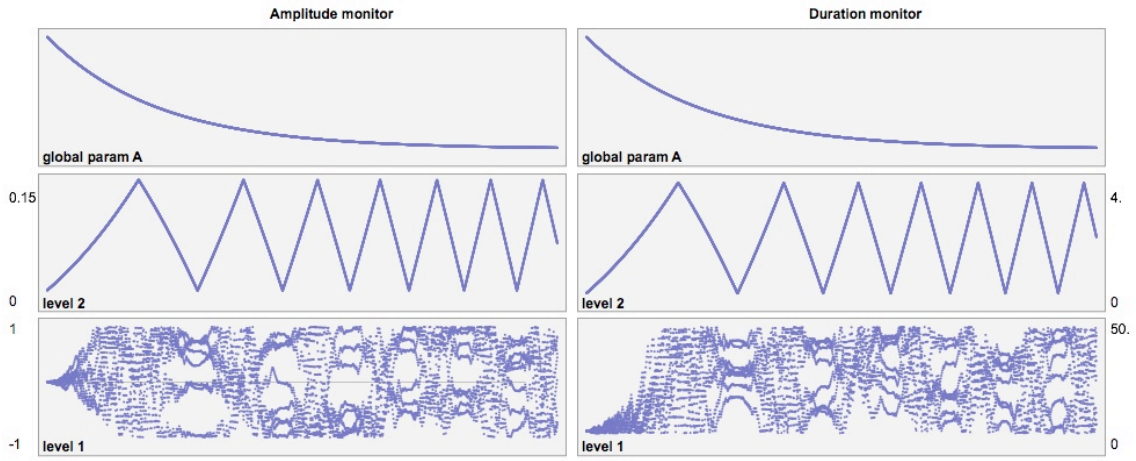

Figure 41: case study 6.1 sequence window

The second sub-case utilizes a random walk whose step or elastic barriers are controlled by a second *random walk function*. This is the dynamic stochastic synthesis type that we have already explored in case study 2. This type is capable of generating dynamic stochastic sound morphologies that are found in the GENDY system of Iannis Xenakis. The global parameter A was set to function mode that controlled the step of the walk function in leve 2. The settings used within the *"DWSS_group*" application for this case study was:

- group length: 8.000 structures
- amplitude global parameter A: function mode with range 0. – 0.01
- amplitude Level 2: walk function with range 0. – 0.05
- amplitude Level 1: walk function with range -1. – 1.
- duration global parameter A: function mode with range 0. – 4.
- duration Level 2: triangle function with range 0. – 5.
- duration Level 1: walk function with range 2. – 50.

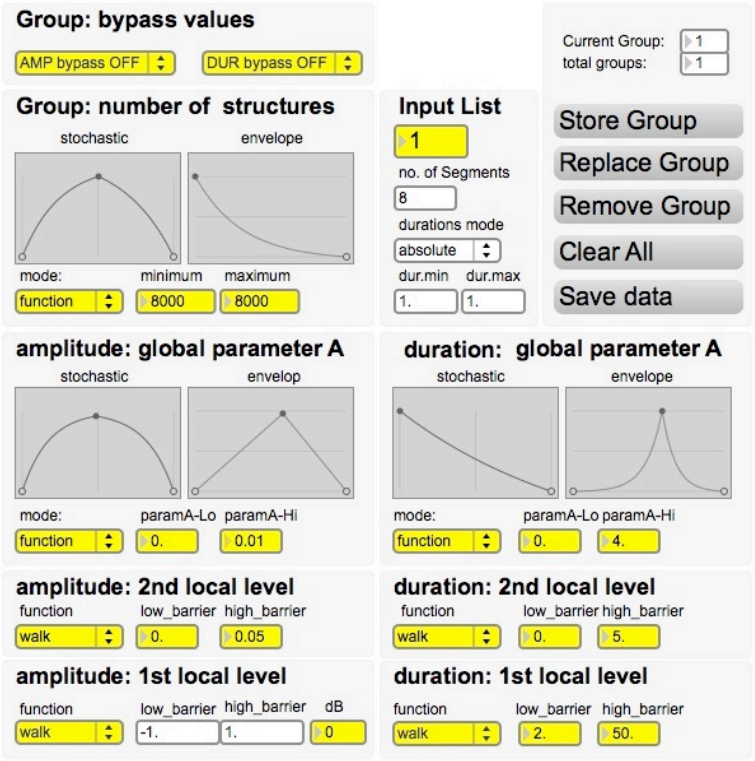

Figure 42: case study 6.2 group window

The behavior of a random walk function controlling a second random walk function can be observed graphically from the corresponding monitor windows of the "*DWSS_sequence*" application.

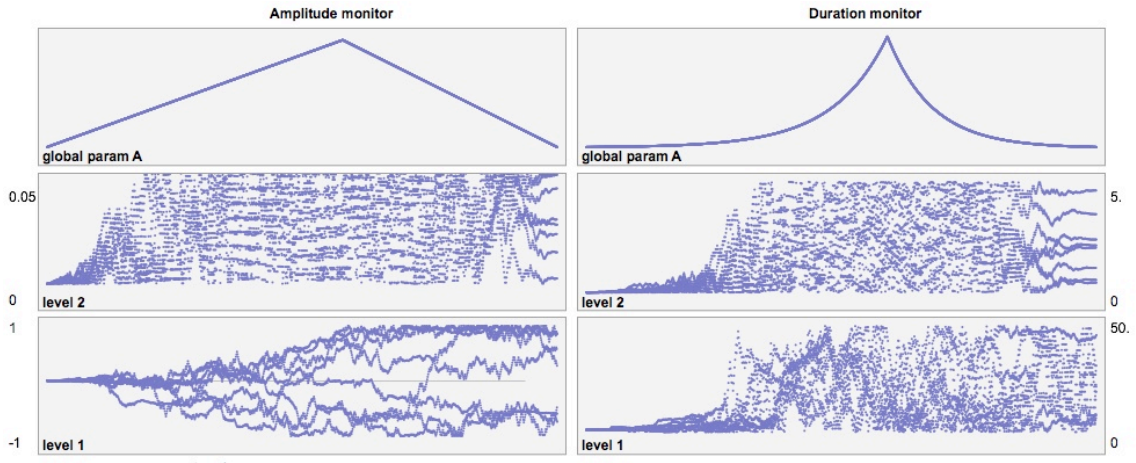

Figure 43: case study 6.2 sequence window

The third sub-case utilizes a random walk whose step or elastic barriers are controlled by a chaotic *logistic map function*. This type is capable of generating dynamic stochastic sound morphologies that are even more unstable or noisy than in the previous case study. The global parameter A was set to function mode that controlled the [r] value of the logistic map in a range that is capable of exploring all the chaotic behavior of the latter. The settings used within the *"DWSS_group*" application for this case study was:

- group length: 8.000 structures
- amplitude global parameter A: function mode with range 2.5 – 4.
- amplitude Level 2: logistic map function with range 0. – 0.2
- amplitude Level 1: walk function with range -1. – 1.
- duration global parameter A: function mode with range 2. – 3.9
- duration Level 2: triangle function with range 0. – 5.
- duration Level 1: walk function with range 2. – 50.

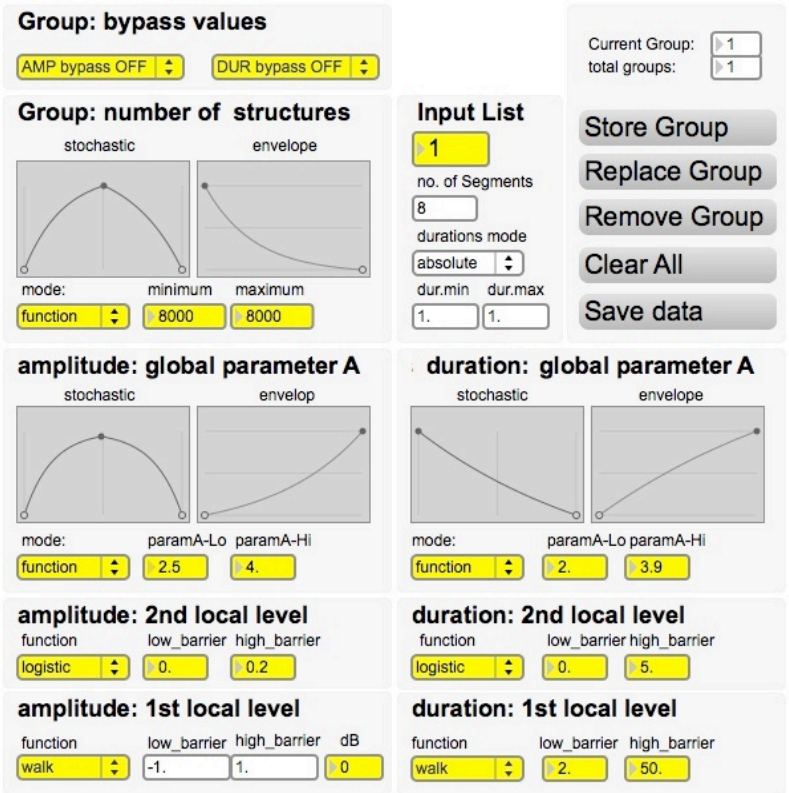

Figure 44: case study 6.3 group window

The behavior of the logistic map function controlling the walk function can be observed graphically from the corresponding monitor windows of the "*DWSS_sequence*" application.

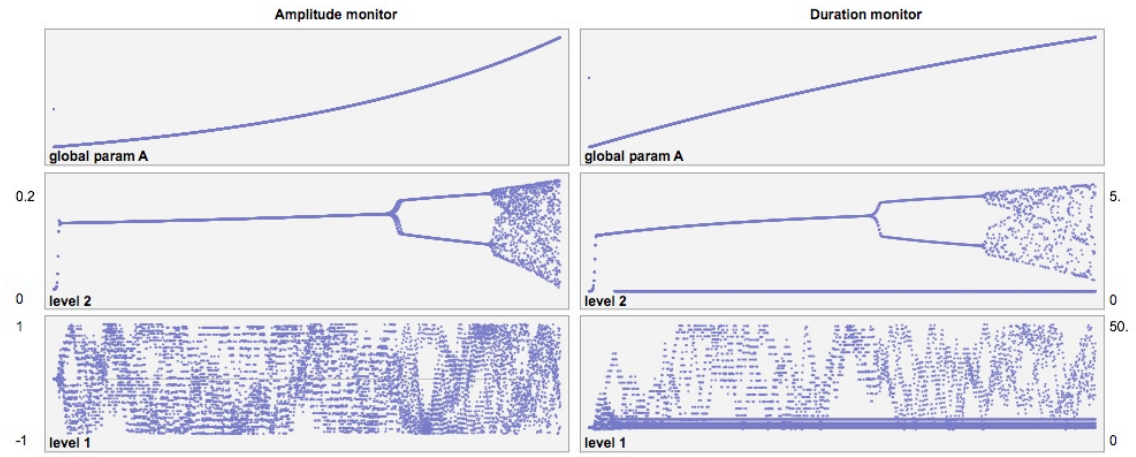

Figure 45: case study 6.3 sequence window

The fourth sub-case utilizes a random walk whose step or elastic barriers are controlled by a chaotic *iterative sine function*. Since the iterative sine function is capable of producing values that range between smooth long oscillations to short oscillations to abrupt jumps, this type is capable of generating dynamic sound morphologies that range between the oscillating features of the triangle function and the noisy feature of the logistic map function. The global parameter A was set to function mode that controlled the [r] value of the iterative sine function in a range that is capable of exhibiting rather smooth oscillations during the beginning. The settings used within the *"DWSS_group*" application for this case study was:

- group length: 8.000 structures
- amplitude global parameter A: function mode with range 3 – 4.

- amplitude Level 2: iterative sine function with range 0. – 0.1
- amplitude Level 1: walk function with range -1. – 1.
- duration global parameter A: function mode with range 3.4 – 3.9
- duration Level 2: iterative sine function with range 0. – 5.
- duration Level 1: walk function with range 2. – 50.

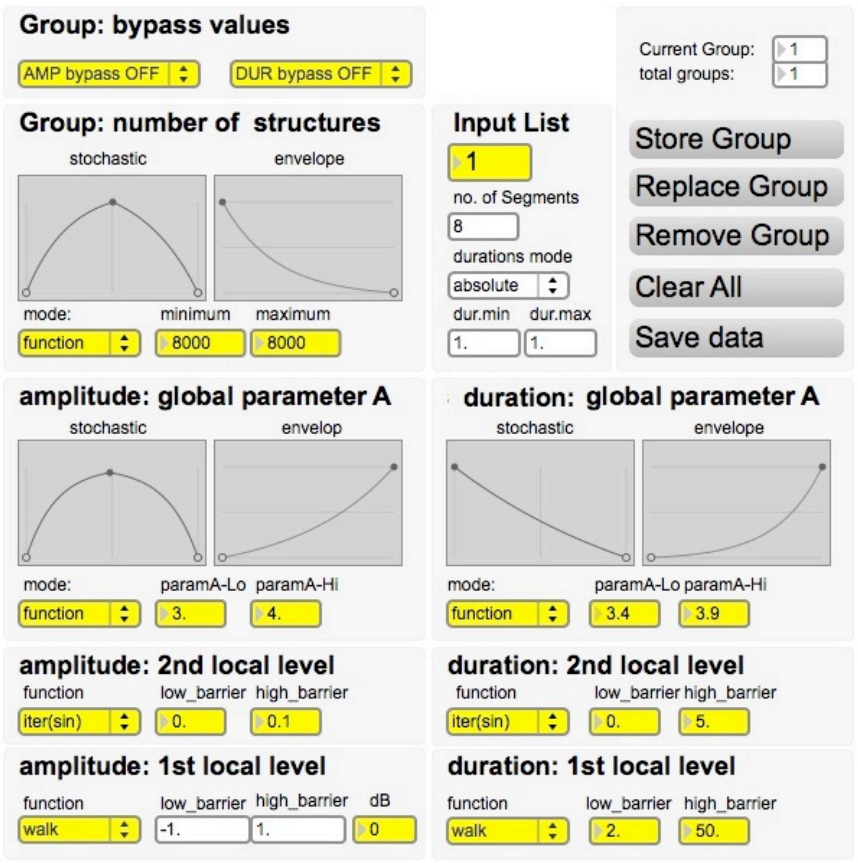

Figure 46: case study 6.4 group window

The behavior of the logistic map function controlling the walk function can be observed graphically from the corresponding monitor windows of the "*DWSS_sequence*" application.

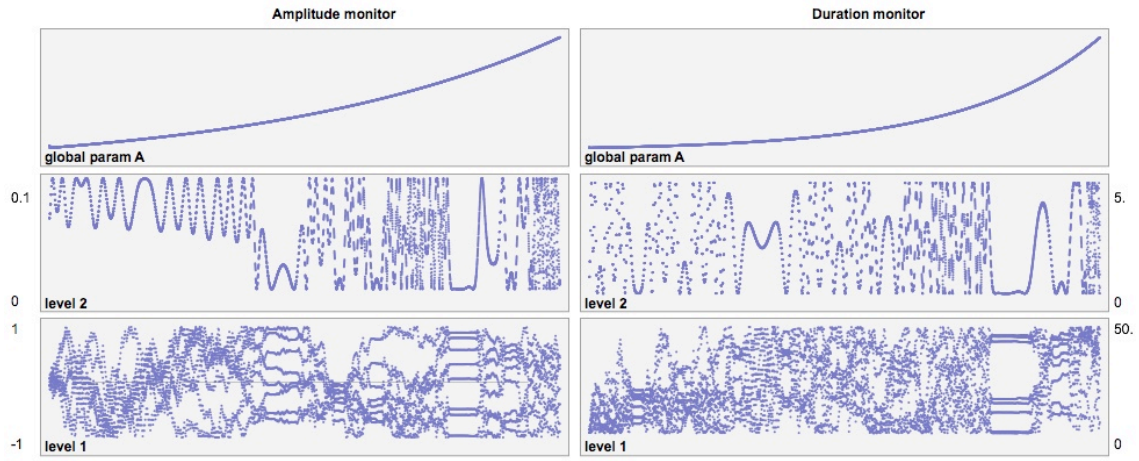

Figure 47: case study 6.4 sequence window

The fifth sub-case utilizes a random walk whose step or elastic barriers are controlled by taking into account the *values of neighbor* segments. This is a typical Cellular automaton behavior. The provided settings produced a behaviour that started with two long oscillations that stabilized the stochastic behavior to two succeeding pitches and eventually led to more complex and noisy patterns. The global parameter A was set to function mode that controlled the weight of the sum of neighbor segments. The settings used within the *"DWSS_group*" application for this case study was:

- group length: 8.000 structures
- amplitude global parameter A: function mode with range 3 – 4.
- amplitude Level 2: iterative sine function with range 0. – 0.1
- amplitude Level 1: walk function with range -1. – 1.
- duration global parameter A: function mode with range 3.4 – 3.9
- duration Level 2: iterative sine function with range 0. – 5.
- duration Level 1: walk function with range 2. – 50.

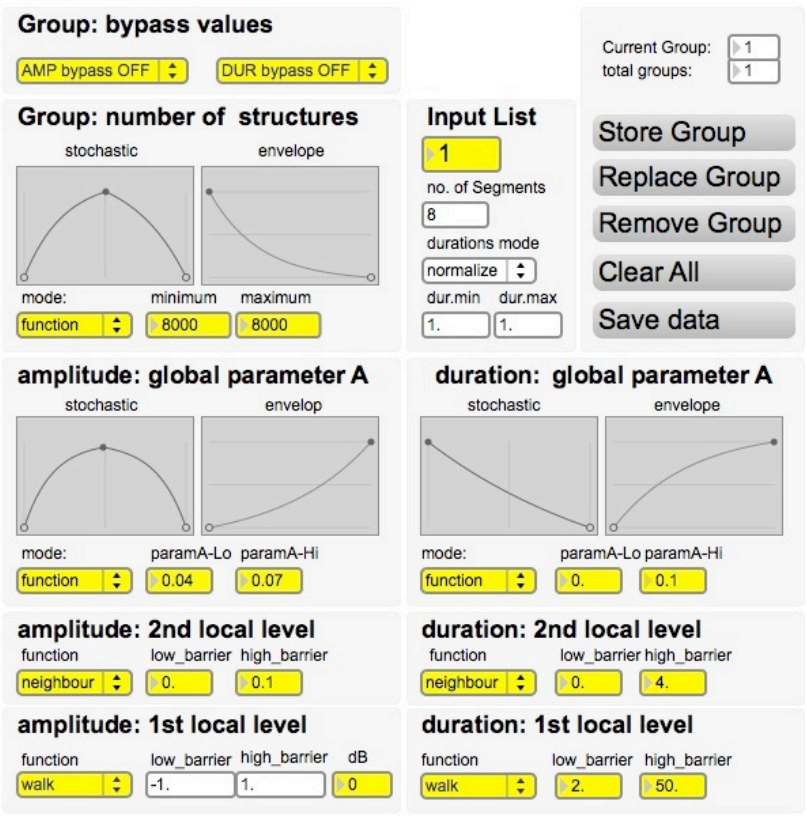

Figure 48: case study 6.5 group window

The behavior of the cellular automaton function controlling the walk function can be observed graphically from the corresponding monitor windows of the "*DWSS_sequence*" application.

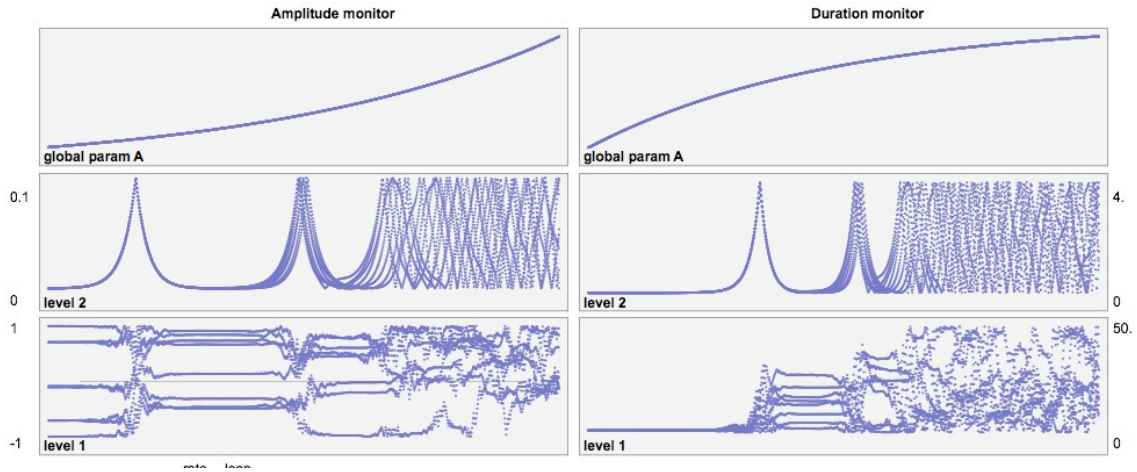

Figure 49: case study 6.5 sequence window

13.7 *Case Study 7: Cellular Automata Sound Synthesis*

In this case study, the aim was to generate with DWSS novel sound objects by utilizing the Cellular Automaton algorithmic process. Although the CA has already been demonstrated, this case study focuses especially on the exhibition of self-organizing behavior that emerges from the interactions between sound segment values.

For this case study we constructed with the “DWSS_generation” one relatively simple structure. This structure consists of nine breakpoints connected with linear link-shapes. We decided to focus the CA activity on both the amplitude and duration breakpoint values. All amplitude values were set at random between the values of 0. and 1.. This initial structure represents also the *initial conditions* of the CA.

Within the “DWSS_group” application we used set the Level 1 to “neighbor” function which takes into account the values of neighbor cells. The Level 2 was not used, so it was set to bypass mode. The *transition rule* was also simple: in every next CA generation, the new cell value is calculated by taking into account the current state of the cell as well as the weighted state of adjacent cells. The global parameter A was set to the function mode which controlled the weight between the values of 0.1 and 0.3. When the calculated next value exceeded the nominal range [0.-1.] then a) it was reflected back by the application of the elastic barriers and b) swap the sign of the weight. The group values used in this case study was:

- group length: 4.000 structures
- amplitude global parameter A: function mode with range 0.1 – 0.3

- amplitude Level 2: bypass
- amplitude Level 1: neighbor function with range -1. – 1.
- duration global parameter A: function mode with range 2. – 10.
- duration Level 2: bypass
- duration Level 1: neighbor function with range 0. – 1.

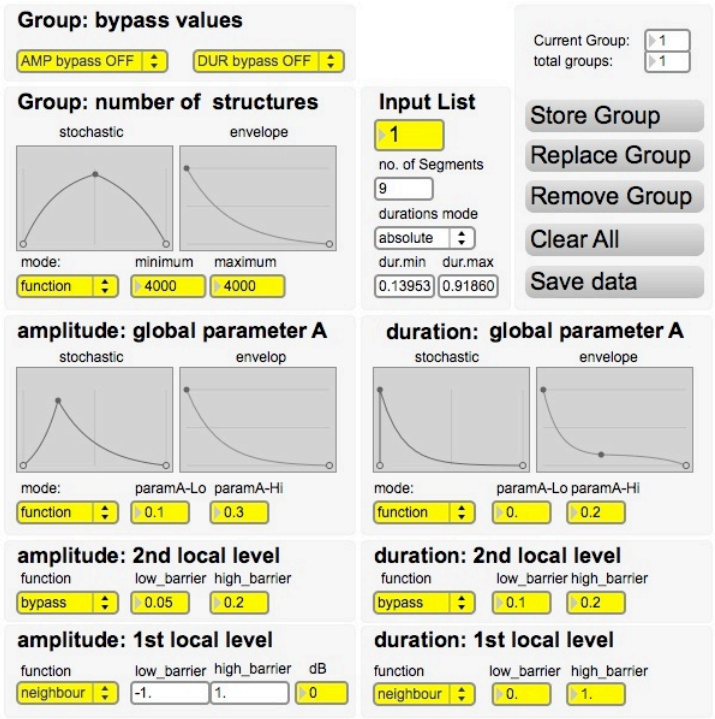

Figure 50: case study 7 group window

With the above transition rule, each segment breakpoint exhibited smooth but unstable modulation changes. The rate of the modulations was not constant but changed continuously according to the interactions of adjacent cells. One factor that was responsible for the rate of the modulation was the weight of the state of the cells. In this case study we explored the generation of a novel sound object by the utilization of the CA mechanism of DWSS.

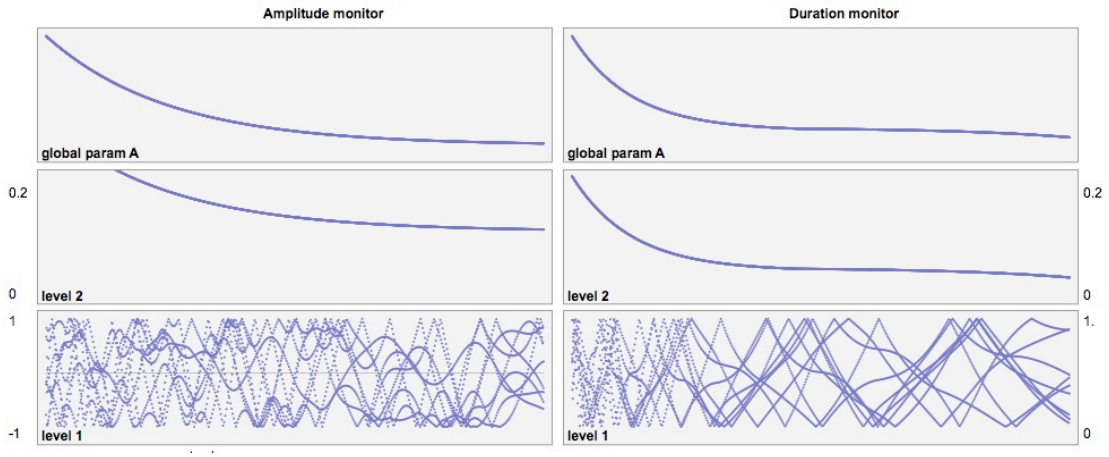

Figure 51: case study 7 sequence window

13.8 *Case Study 8: Dynamic Microrythmic Morphologies*

The aim of this case study was to investigate the capabilities of DWSS to generate dynamic microrythmic morphologies. By the term microrythmic morphologies we define rhythmic properties of a sound object that occur within the microsonic domain. In other words, a special category of rhythmic structures where the rhythmic elements have very short time durations, usually lasting up to 40 milliseconds.

DWSS is capable of generating sequences that are the assemblange of groups sound structures. Each group may be considered as an individual microrythmic element. Consequently, each sequence may be considered as the framework for the assemblage of microrythms. Microrythms are another morphological emergence within DWSS, as the result of the application of dynamic algorithmic processes in the assemblage of structures into group and groups into sequences.

The “*DWSS_sequence*” application provides four categories of assembling procedures:

- oscillating functions (phasor & triangle)
- random walk
- chaotic functions (logistic map)
- L-system

First, for the purposes of this case study we defined the properties for four group structures. We used the DWSS_group” application for the generation of four group structures. The same four groups used for the construction of different sequences that investigate a variety of microrythmic assemblages by the application of phasor, random walk, logistic map and L-system functions. The four identical group structures provided the basis of common sound morphologies that would help for the identification of the microrythmic properties of the four functions in use. For each group we provided the same functions for both the amplitude and duration sections. The length for all groups was chosen by a stochastic distribution within the range of 4-8 structures. The generated sequence was set for all the examples to 5.000 segments.

The first group is distinguished by the stochastic evolution of the segment values. Both Level 2 & Level 1 uses walk functions. Particularly, the group values used in the first group was:

- group length: stochastic mode with range 4 - 8 structures
- amplitude global parameter A: stochastic mode with range 0. – 0.1
- amplitude Level 2: walk function with range 0. - 0.2
- amplitude Level 1: walk function with range -1. – 1.

 duration global parameter A: stochastic mode with range 0. – 0.05

 duration Level 2: walk function with range 0. - 0.1

 duration Level 1: walk function with range 0. – 1.

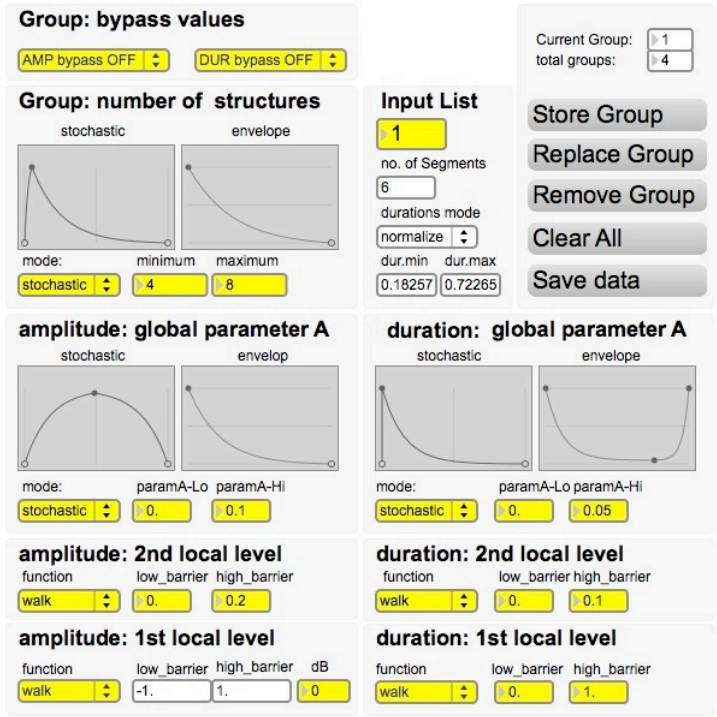

Figure 52: case study 8 group 1 window

The second group is distinguished by cyclic evolution of the segment values. Both Level 2 & Level 1 uses triangle functions. Particularly, the group values used in the first group was:

 group length: stochastic mode with range 4 - 8 structures

 amplitude global parameter A: function mode with range 30. – 50.

 amplitude Level 2: triangle function with range 10. – 20.

 amplitude Level 1: triangle function with range -0.22. – 0.22

 amplitude global parameter A: function mode with range 30. – 50.

 amplitude Level 2: triangle function with range 50. – 100.

 amplitude Level 1: triangle function with range -1. – 1.

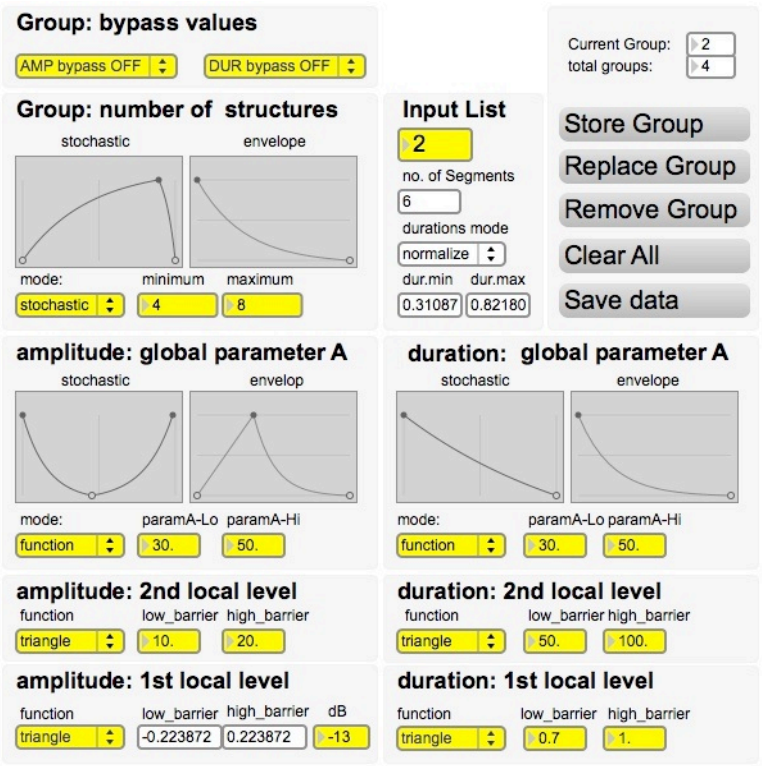

Figure 53: case study 8 group 2 window

The third group is distinguished by the stochastic evolution of the segment values controlled by cyclic function. Particularly, the group values used in the first group was:

- group length: stochastic mode with range 4 - 8 structures
- amplitude global parameter A: stochastic mode with range 30. – 50.
- amplitude Level 2: triangle function with range 0. - 0.1
- amplitude Level 1: walk function with range -1. – 1.
- duration global parameter A: stochastic mode with range 30. – 50.
- duration Level 2: triangle function with range 0. - 0.1
- duration Level 1: walk function with range 0. – 1.

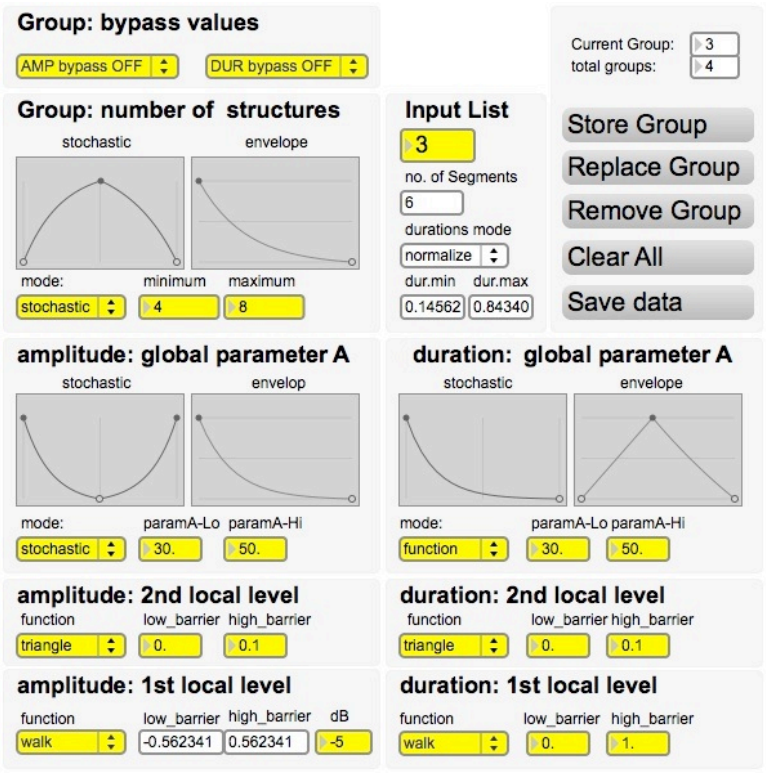

Figure 54: case study 8 group 3 window

The forth group is distinguished by chaotic evolution of the segment values. Both level 2 & level 1 uses chaotic functions. Particularly, the group values used in the first group was:

- group length: function mode with range 4 - 8 structures
- amplitude global parameter A: function mode with range 3.3 – 3.5
- amplitude Level 2: logistic map function with range 3.1 – 3.8
- amplitude Level 1: iterative sine function with range -0.89. – 0.89
- amplitude global parameter A: function mode with range 3. – 3.1
- amplitude Level 2: logistic map function with range 3. – 3.1
- amplitude Level 1: iterative sine function with range -1. – 1.

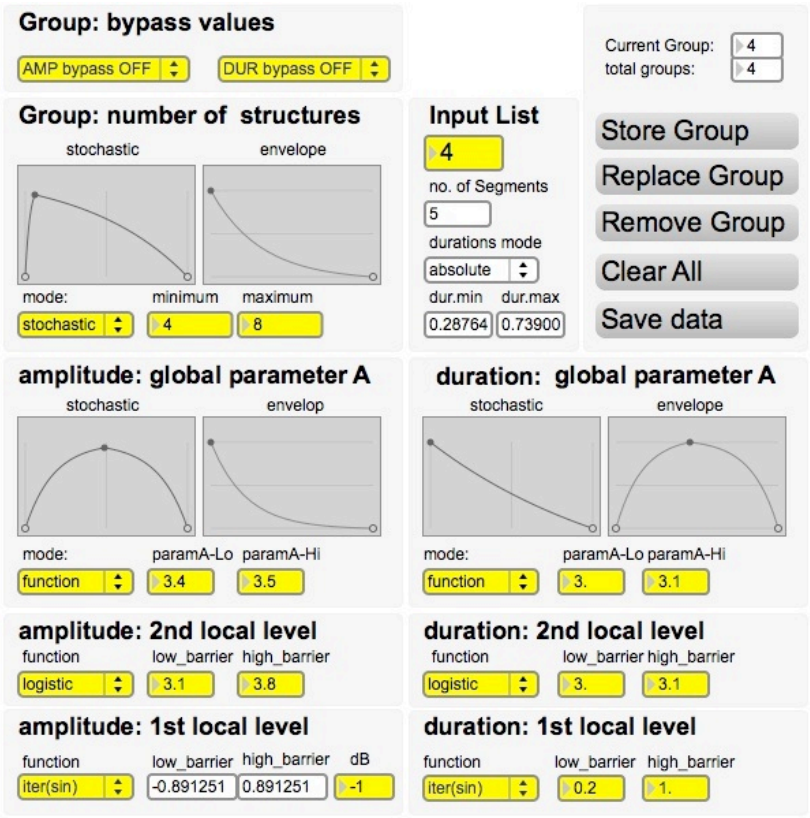

Figure 55: case study 8 group 3 window

The first example of this case study is *dynamic cycling microrythm*. The microrythmic properties of the sound structure was generated with the help of the phasor function. The oscillating functions in general and the phasor function in particular performs cyclic scan through the microrythmic elements, therefore is the most appropriate for the generation of repetitive microrythms.

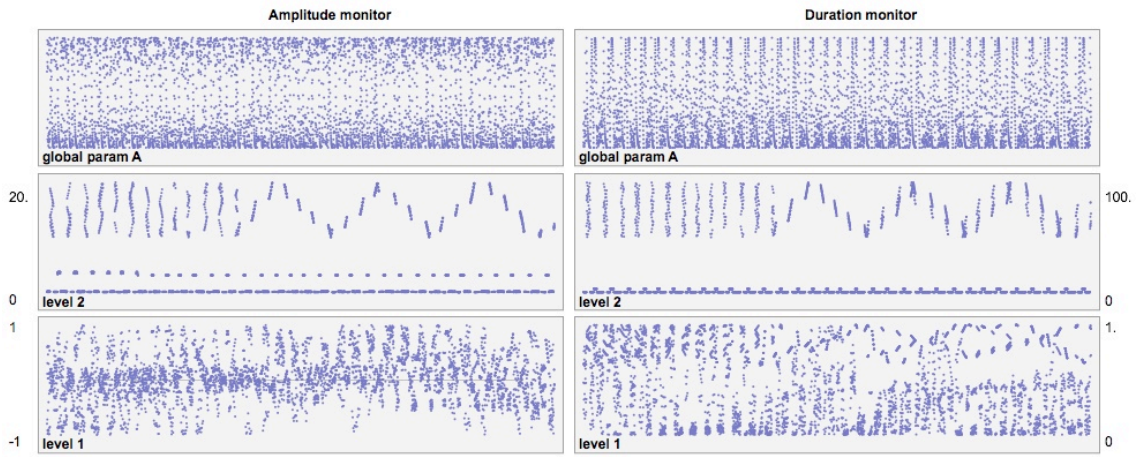

Figure 56: case study 8 sequence 1 window

The second example of this case study is *dynamic stochastic microrythm*. The microrythmic properties of the sound structure was generated with the help of the walk function. The random walk functions performs stochastic scan through the microrythmic elements. The random walk generates various kinds of stochastic microrhythms according to the function and the step. For this case study we defined 0.2 as the random step value.

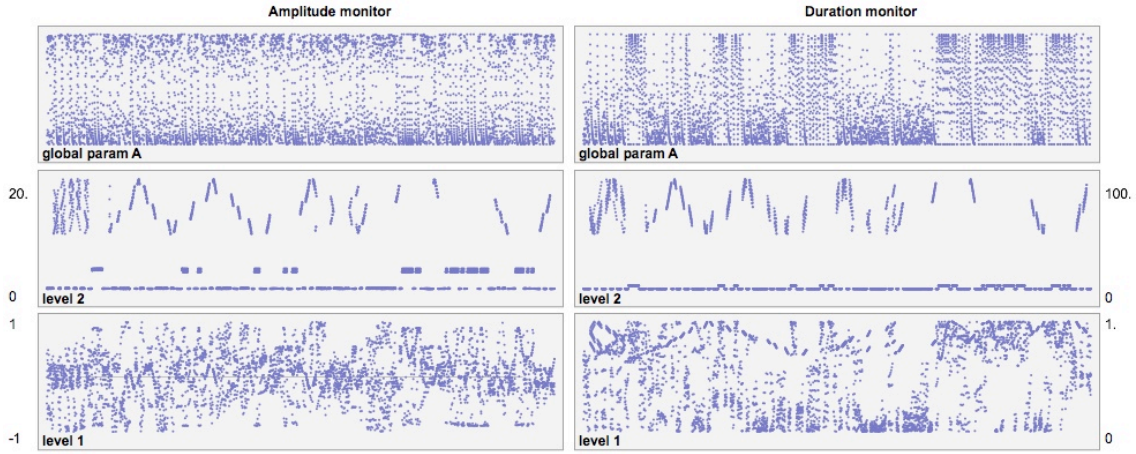

Figure 57: case study 8 sequence 2 window

The third example of this case study is *dynamic chaotic microrythm.* The logistic map chaotic function generates various dynamic microrythmic morphologies according to the behaviour of the attractor: from monotonic microrythms (fixed-point), to microrythmic repetitions (limit-cycle) to complex chaotic behaviour (strange). For this case study we defined 3.89 as the initial condition for the chaotic function.

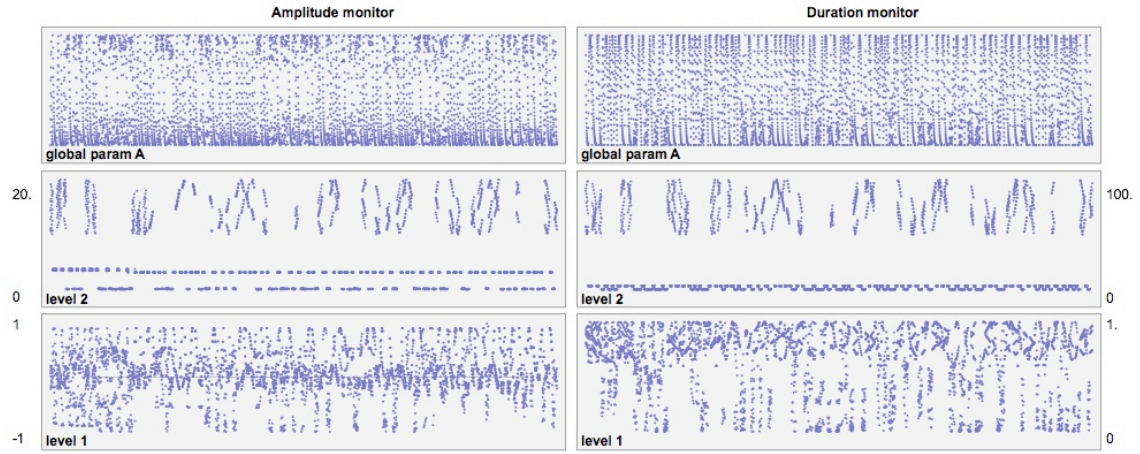

Figure 58: case study 8 sequence 3 window

The fourth example of this case study is *dynamic grammar microrythm.* The L-system is capable of generating various evolving microrythms according to the provided production rule. For this case study we defined the production rule: (A→BD), (B→AC), (C→BAD) and (D→A).

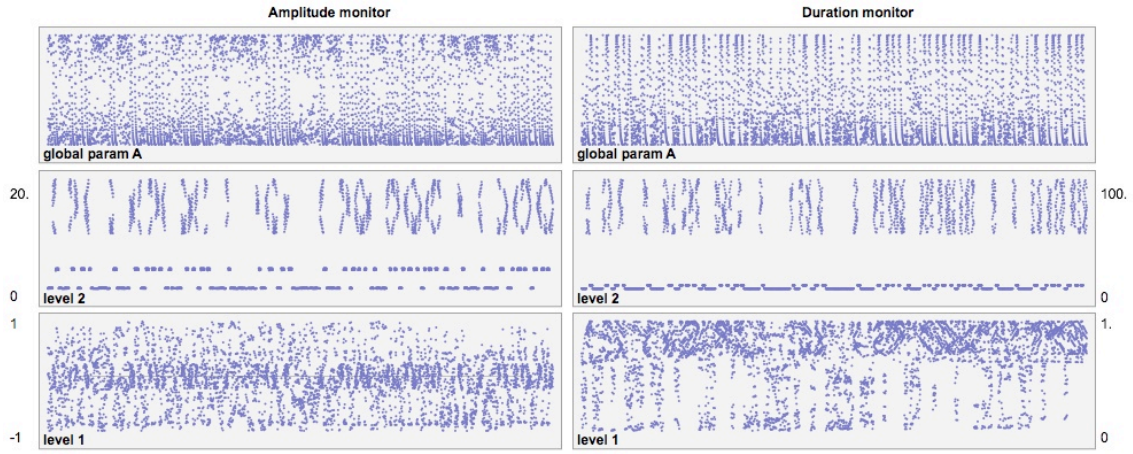

Figure 59: case study 8 sequence 4 window

14 Conclusions

14.1 *Contributions to knowledge*

The contributions to knowledge introduced in this Thesis are presented below:

14.1.1 The Extended Waveform Segment Synthesis (EWSS) model

This Thesis introduces (chapter 14) a novel non-standard synthesis paradigm that utilizes dynamic algorithmic models for the generation of sound objects. This paradigm, the Extended Waveform Segment Synthesis (EWSS) model, provides a generalized concept that:

- Incorporates all existing waveform segment synthesis techniques into a single model.
- Constitutes the basis for a novel non-standard synthesis model proposed in this Thesis: Dynamic Waveform Segment Synthesis (DWSS).
- Provides a solid framework for further future research.

Since in EWSS, a waveform is constructed by assembling blocks of amplitude fluctuations with very short durations in the scale of microseconds, this Thesis provides definitions for the notions of *segment*, *breakpoint and*, *link*.

Additionally, this Thesis proposes a hierarchical approach in sound construction within the paradigm of EWSS, from small waveform fragments towards complete sound objects. EWSS actually proposes four different levels: *segment*, *structure, group, sequence* and, *sound object* level.

14.1.2 The Dynamic Waveform Segment Synthesis (DWSS) model

This Thesis describes the implementation of the concepts and ideas of EWSS in a computer music environment. These concepts are implemented in Dynamic Waveforms Segment Synthesis (DWSS), a novel model for non-standard synthesis model. DWSS applies the notions of *segment*, *structure*, *group*, and *sequence* and utilizes a combination of dynamic algorithmic generative and transformative models. These models are: list generation, list permutation, tendency mask, trigonometric functions, stochastic functions, chaotic functions cellular automata and grammars.

The application of CA for the transformation of structures is one of the main contributions of this Thesis. One or more CA, with simple or complex rules, are responsible for the transformation of the *structures* into *groups*. Two independent Cellular Automata (CA) operating in parallel controls the variation of the amplitude and duration breakpoint value of the structure. In order to fulfill the requirements of floating point arithmetic's, of DWSS is driven by the special category of *continuous CA.* DWSS is using a variety of transition rules that incorporate chaotic, stochastic, trigonometric algorithmic models.

The utilization of CA along with other dynamic algorithmic models constitutes DWSS capable of generating sound objects that feature a variety of sound morphologies as well as exhibiting the phenomenon of emergence of 2^{nd} order sonorities.

14.1.3 Extensive and critical overview on the concepts of: Microsound, Algorithmic Composition and, Non-Standard Synthesis

This Thesis provides an extensive and critical overview on a number of fundamental concepts on the subject of this research.

First, we investigated the concept of *microsound* (chapter 2) and the *hierarchy of the time-level scale* in relation to music composition. In this respect, we discussed (i) the *sample level*, (ii) the *micro level* along with *synthesis and transformation* approaches, (iii) the *sound object level* along with the concepts of *spectromorphology* and *reduced listening*. Moreover, we further examined the phenomenon of *sonic emergence* (chapter 8) as well as, *the dialectics of music form and the sonic material* (chapter 9).

Second, we discussed the concept of *algorithmic composition (*chapter 3). We critically examined some principle definitions of the term that has been historically used to define usually overlapping and occasionally identical concepts in the field. We discussed different approaches from various composers and musicologists and we focused in the concept of "totalistic algorithmic composition". In this respect, we researched and provided the *historical and theoretical foundations* of the term (chapter 4).

Third, we reviewed the concept of *Non-Standard Synthesis* (chapter 7) and provided important historical definitions. For his purposes we discussed the concepts of "*two cultures of computer music*" (chapter 5), the heretical approach

to computer music (chapter 6) and the *idiomatic character* of non-standard synthesis (section 7.2).

14.1.4 Survey of Computer Models that are utilized in contemporary musical creativity

This Thesis introduces (chapter 11) fundamental *algorithmic models* that are utilized in contemporary composition and synthesis of sound. Most of these models provide the basis for and are applied in Dynamic Waveform Segment Synthesis proposed in this Thesis. The models surveyed are: *permutations* (section 11.2)*, stochastic – probabilities* (section 11.3.1)*, random walk* (section 11.3.2)*, Markov chain* (section 11.3.3), *chaos* (section 11.4.1)*, fractals* (section 11.4.2)*, grammars* (section 11.5)*, Lindenmayer systems* (section 11.6) and *cellular automata* (section 11.7). These models were discussed along with important musical and compositional applications.

14.1.5 Survey of Non-Standard Synthesis approaches and systems

This Thesis introduces (chapter 11) diverse non-standard synthesis approaches and surveys a number of systems that implements them. We discussed the following systems: Iannis Xenakis's *Dynamic Stochastic Synthesis* (13.1)*, GENDY* (13.1.1) and recent variations (13.1.2), G.M. Koenig's *SSP* (13.2), Paul Berg's *PILE* (13.3), Herbert Brün's *SAWDUST* (13.4), and, Arun Chandra's *TrikTraks* and *Wigout* (13.5).

14.1.6 A plausible definition for "Algorithmic Composition"

During the discussion of the concept of algorithmic composition and the critically examination of some principle definitions of the term, we concluded the term does not have a generally accepted definition. This Thesis regards that algorithmic procedures may be applied at a variety of compositional levels and that it is an inextricable amalgam of concepts, procedures and human choices regardless the use of the computer. In this respect, this Thesis proposes a plausible definition (chapter 3):

> *"Algorithmic composition consists of musical concepts that are formalized and employed by the composer in rules and procedures that generate elements, parts or the whole musical work."*

14.1.7 A proposal for the Term "Totalistic Algorithmic Composition"

During the discussion of the term algorithmic music, we investigated the concept of algorithmic operations that are capable of generating entire compositions from the whole macro-structure down to the micro-sonic detail. Although other composers and researchers have used terms like "integrity" [Laske 1981], "pure" [Ariza 2005] or "rigorous" algorithmic composition [Hoffman 2009] this Thesis considers as more appropriate and proposes the term *totalistic. We state in section 3.1:*

> *We call "totalistic" the algorithmic music approach that utilizes computer procedures for the generation of the entire composition up to its finest detail.*

14.1.8 Categorization of Non-standard Synthesis into discrete aspects

This Thesis regards that non-standard synthesis provides a very idiomatic paradigm of music composition whose very essence is derived from the notion of machine computability itself. Questioning the idiomatic qualities of non-standard synthesis, this Thesis proposes (section 7.2) four discrete aspects:

formal: each non-standard synthesis system proposes a novel approach in sound modeling. The algorithm is an abstract conception and does not rely on any pre-existing theory or higher order model. Any non-standard formalization is a new proposal for a unique sound model.

sonical: the sound world of each non-standard synthesis is highly distinguishable and non-standard systems are capable of generating novel sound structures of unheard-off sounds.

personal: non-standard synthesis define a very personal artistic idiom. Each non-standard approach is based on a specialized formalization that is capable of generating a sonic world that eventually characterizes the artistic identity of its designer artist.

machine oriented: non-standard synthesis is idiomatic to specific machine or at least that was the case for the early examples.

14.1.9 Demonstrative Contribution with a number of Case Studies

This Thesis presents (chapter 17) eight case studies that reveal the capabilities of Dynamic Waveform Segment Synthesis (DWSS) into generating particular sonic morphologies.

On the one hand, we provide a number of case studies that indicate how DWSS is capable of generating sound objects that are characteristic of other approaches and that incorporate basic features of other microsound synthesis techniques: *Soundfile Segmentation and Resynthesis* (section 17.1), *Dynamic Stochastic Synthesis* which is the basis of Xenakis's GENDY system (section 17.2), *Iterated Nonlinear Functions* which is the basis of Di Scipio's Iterated Nonlinear Functions (section 17.3), *Oscillating functions & Tendency Masks* which are the basis of Koening's SSP and Brun's SAWDUST systems (section 17.4), *Graphics and Grammars* which is the basis of Manousakis;s "Non-standard Sound Synthesis with L-Systems" (section 17.5)

One the other hand, we provide a number of case studies disclosing the novelty of DWSS and how it can generate new sound objects featuring morphologies that belong to the heretical currents of contemporary computer music creativity: *Dynamic Sound Synthesis* (section 17.6), *Cellular Automata Sound Synthesis* (section 17.7) and, *Dynamic Microrythmic Morphologies* (section 17.8)

However, since DWSS belongs to the non-standard synthesis approaches, the provided case studies are but a subset of the potential sound morphologies. Non-standard synthesis requires a lot of laborious experimentation and recompense with the discovery of new sonic territories.

14.2 *Recommendations for Future Work*

Although DWSS provides a solid implementation of non-standard synthesis with dynamic models, further improvements are recommended for future work and expansion of the system.

14.2.1 Expansion and Interconnection between the hierarchical construction levels

In the current implementation of DWSS, sound construction takes a hierarchical approach that incorporates four different levels: segment, structure, group and, sequence.

In a future expansion of the system, the incorporation of more levels would provide further structural detail in the assemblage of smaller units into larger entities and the final sound object. One relatively simple solution would be the addition more levels above the sequence level with similar algorithmic procedure(s).

Another feature that might provide interesting microsonic results would be the interconnection between the different hierarchical levels. For example, the algorithmic output of a lower level (eg. group) might control and affect the construction of a higher level. This would transform DWSS into a large recurrent system where the hierarchy is questioned and the construction levels become interdependent. Since recurrent systems are capable of exhibiting different types of chaotic behavior, this option may open new possibilities both in non-standard waveform construction as well as the generation of novel sound morphologies and sonorities.

14.2.2 Detailed investigation of CA transition rules

In the current implementation of DWSS, Cellular Automata play an important role in the final construction of the sound object. One important aspect of CA that plays significant role on the evolution of the system is the transition rule. In this Thesis we have explored only a limited number of simple transition rules. A further experimentation with either different simple rules or with more complex rules would be proved useful for the further investigation of the role of the CA in sound construction as well as the systematization of the algorithmic procedure.

14.2.3 Grammars with complex rules

In DWSS, grammars may be utilized for the initial segment list construction. Since the initial lists, with the relevant optimizations of the system, may highly affect the final construction of the sound object, the use of grammars may play an important role. Therefore, another important expansion of the system would be the potentiality of setting more complex grammars's rules. This option would take DWSS closer to the "Non-standard Sound Synthesis with L-Systems" approach of Stelios Manousakis.

14.2.4 Improved sound segmentation algorithm

In DWSS sound segmentation is an important feature since it incorporates the utilization of sound samples or sonification data as the starting point for non-standard sound transformation. In DWSS we defined a simple segmentation algorithm that identifies zero-points as well as intermediate points with the minimum and maximum values. The current algorithm segments each wavecycle of a waveform at best into four segments. A possible improvement of

the algorithm would identify more segmentation points and therefore would produce a larger number of link-shapes.

14.2.5 Graphical Interactive User Interface

The implementation of a graphical user interface (GUI) would highly improve the user experience of the system. A GUI would help the user to easily comprehend the hierarchical structuring levels as well as the involved algorithmic procedures. Additionally, a GUI would provide the user with all the necessary tools to navigate more easily the system and control in an intuitive way its parameters. Moreover, a GUI gives important feedback on the current state of the system as well as the sound output.

14.2.6 Real-time operation

Finally, the real-time operation of DWSS would improve its functionality and provide the user another option to investigate the various algorithmic features of the system and experiment more thoroughly with the generated sound objects. Real-time operation would require the transfer of the system to another programming environment, probably C or C++. This would improve the processing speed on operations carried in the sound sample level.

Apendix I – record of activities

1. Papers

Valsamakis N., E.Miranda (2005), "Extended Waveform Segment Synthesis, o Nonstandard synthesis model for microsound composition", Sound & Music Computing 2005, Salerno, Italy

Valsamakis N., E.Miranda (2005), "Iterative Sound Synthesis by means of Cross Coupled Oscillators", Digital Creativity. 16(2):79-92, 2005, Routledge – Taylor & Francis Group.

2. Public Performances

16-18 March 2012. Di.P.Art. Festival. Kodra, Kalamaria, Thessaloniki, Greece

2-4 December 2011. Electroacoustic Music Days 2011. Dept. Of Music Technology & Acoustics, TEI of Crete, Greece

9 November 2011. Hellenic Association for Electroacoustic Music. Music, Technology and Innovation Research Center, De Monfort University, UK

29-31 October 2010. Electroacoustic Music Days 2010. Dept. Of Music, Ionian University, Greece

13-16 November 2009. Electroacoustic Music Days 2009. Dept. Of Sound Technology & Acoustical Instruments, TEI of Ionian Islands, Greece

15-21 June 2009. Electroacoustic Composition "Voices of the Desert" (2009) for the performance/installation "Mystical Illusion" by Doris Hakim. Athens Fringe Festival 2009, Technopolis, Gazi, Athens

25-28 October 2008. Electroacoustic Music Days 2008. Dept. Of Music Technology & Acoustics, TEI of Crete, Greece

21 June 2008. European Music Day. Xia-Mass & Municipality of Chania, Chania, Greece

21-24 November 2007. Electroacoustic Music Days 2007. Dept. Of Sound Technology & Acoustical Instruments, TEI of Ionian Islands, Greece

11-13 July 2007. Sound and Music Computing International Conference 2007 (SMC07). University of Athens, Ionian University, SMC, Lefkada, Greece

3 March 2006. Cage Mix. Appolo 39 Bac Bar, IDAT, Plymouth Univercity, UK

28-30 October 2005. Electroacoustic Music Days 2005. Dept. Of Music Technology & Acoustics, TEI of Crete, Greece

8 October 2005. 2nd Marathon of Electroacoustic Music. Goethe Institute, Athens, Greece.

3 July 2005. SYNC Festival. Technological & Educational Park, Lavrio, Greece

24 March 2005. Electronics in Action. Small Music Theater, Athens, Greece

26 February 2005. Peninsula Arts Contemporary Music Weekend. University of Plymouth, Plymouth, UK

3. Seminar Presentations

Valsamakis, N. 2005. "Dynamic Sound Synthesis", Postgraduate research seminar, Plymouth University, UK

Valsamakis, N. 2005. "Extended Waveform Segment Synthesis". Paper presentation in Sound & Music Computing 05, Salerno, Italy.

Valsamakis, N. 2004. "Elica, real-time granular synthesizer". Postgraduate research seminar, Plymouth University, UK

Valsamakis, N. 2004. "Non-Standard Waveform Synthesis for Microsound Composition", Postgraduate research seminar, Plymouth University, UK

Apendix II – sound examples

The following sound examples are provided in as separate cd-rom.

All sound examples are 44.100Hz sampling rate, 16 bit

example 2.aiff

Case Study 2: Dynamic Stochastic Synthesis (chapter 13.2)

A typical sound produced by the GENDY system of I.Xenakis Both segment amplitudes and durations are controlled by random walks (duration 00:05)

example 3.aiff

Case Study 3: Iterated Nonlinear Functions (chapter 13.3)

A typical sound produced by the chaotic IFS system of Agostino di Scipio. The segment amplitudes are controlled by iterative sine function. Since IFS operates at the sample level, all segment durations are 2 samples long, thus eventually lasting only one sample (duration 00:05)

example 4.aiff

Case Study 4: Oscillating functions & Tendency Masks (chapter 13.4)

An audio example where both segment amplitude and segments are controlled by triangle oscillating functions. This examples features Amplitude Modulation & Frequency Modulation (duration 00:07)

example 5.aiff

Case Study 5: Sound Synthesis with graphics & grammars (chapter 13.5)

A sound example that features sound synthesis by the help of a L-system. Two group of structures alternating rapidly (5-10 wavecycle repetition each). The microrythm is generated according to the production rule of the L-system (duration 00:05)

example 6_1.aiff

Case Study 6a: Dynamic Sound Synthesis (chapter 13.6)

In this example a random walk that is responsible for the generation of segment amplitude and duration values is controlled by triangle oscillating functions. This process generates oscillating morphologies that range between a static sound -

when the trigonometric function outputs zero - and full stochastic behavior - when the trigonometric function outputs the maximum value - (duration 00:04)

example 6_2.aiff

Case Study 6b: Dynamic Sound Synthesis (chapter 13.6)

In this example a random walk that is responsible for the generation of segment amplitude and duration values is controlled by a second random walk. This is another typical sound produced by the GENDY system of I.Xenakis (duration 00:02)

example 6_3.aiff

Case Study 6c: Dynamic Sound Synthesis (chapter 13.6)

In this example a random walk that is responsible for the generation of segment amplitude and duration values is controlled by a chaotic logistic map.This process is capable of generating dynamic stochastic sound morphologies that are even more unstable than in the previous example (duration 00:02)

example 6_4.aiff

Case Study 6d: Dynamic Sound Synthesis (chapter 13.6)

In this example a random walk that is responsible for the generation of segment amplitude and duration values is controlled by a chaotic iterative sine function. Since the iterative sine function is capable of producing values that range between smooth long oscillations to short oscillations to abrupt jumps, this type is capable of generating dynamic sound morphologies that range between the oscillating features of the triangle function and the noisy feature of the logistic map function (duration 00:04)

example 7.aiff

Case Study 7: Cellular Automata Sound Synthesis (chapter 13.7)

In this examples the segment amplitude and duration values are controlled according to the values of neighbor segments. This is a classic Cellular Automaton procedure. With the chosen transition rule the sound exhibits smooth but unstable modulations (duration 00:06)

example 8_1.aiff

Case Study 8a: Dynamic Microrythmic Morphologies (chapter 13.8)

This example features four group structures alternating repeatedly and producing cycling mycrorythmic morphologies (duration 00:08)

example 8_2.aiff

Case Study 8b: Dynamic Microrythmic Morphologies (chapter 13.8)

This example features four group structures alternating with the help of a random walk and producing stochastic mycrorythmic morphologies (duration 00:09)

example 8_3.aiff

Case Study 8c: Dynamic Microrythmic Morphologies (chapter 13.8)

This example features four group structures alternating with the help of the chaotic logistic map and producing chaotically unstable mycrorythmic morphologies (duration 00:10)

example 8_4.aiff

Case Study 8d: Dynamic Microrythmic Morphologies (chapter 13.8)

This example features four group structures alternating with the help of an L-system and producing unstable mycrorythmic morphologies that evolve according to the provided production rule (duration 00:10)

Apendix III – software: MaxMSP patches

The following MaxMSP patches are provided in a separate cd-rom:

DWSS_storage.maxpat
DWSS_construction_061.maxpat
DWSS_transformation_033.maxpat
DWSS_group_34.maxpat
DWSS_sequence_052.maxpat
DWSS_synthesis_02.maxpat

The following MaxMSP externals where used in the provided patches:

lp.bernie.mxo
lp.poppy.mxo
lp.scampf.mxo
lp.scampi.mxo
lp.shhh.mxo
from the Litter Power package by Peter Castine
http://www.bek.no/~pcastine/litter/

gen10.mxo
from the PeRColate by Dan Trueman and R. Luke DuBois
http://music.columbia.edu/percolate/

Apendix IV – software code (MaxMSP patches screenshots)

1) DWSS_storage

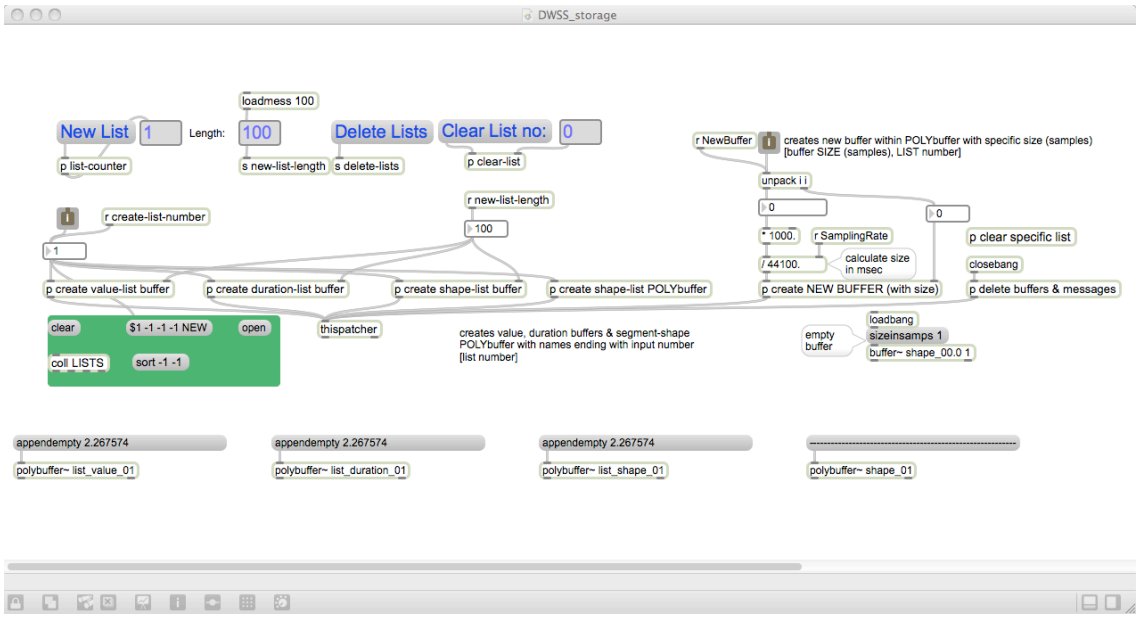

Figure 60: DWSS_storage

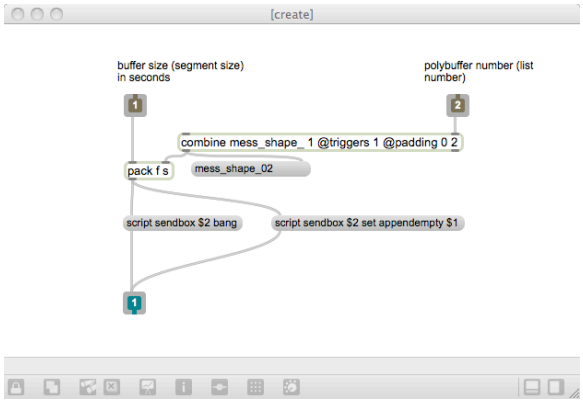

Figure 61:p create NEW BUFFER (with size)

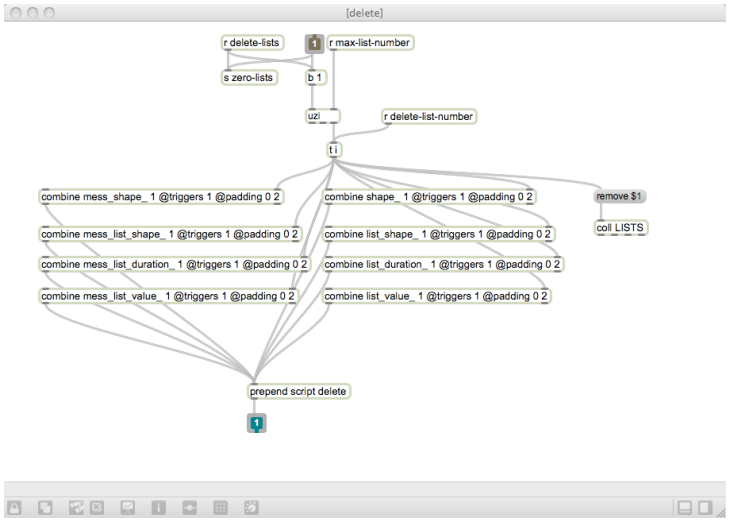

Figure 62: p delete buffers & messages

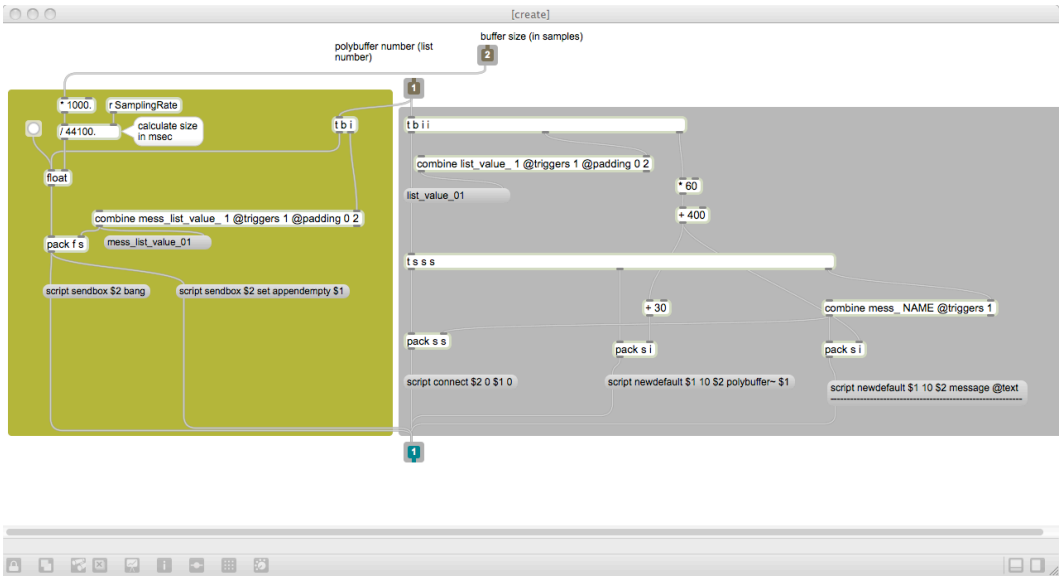

Figure 63: p create value-list buffer

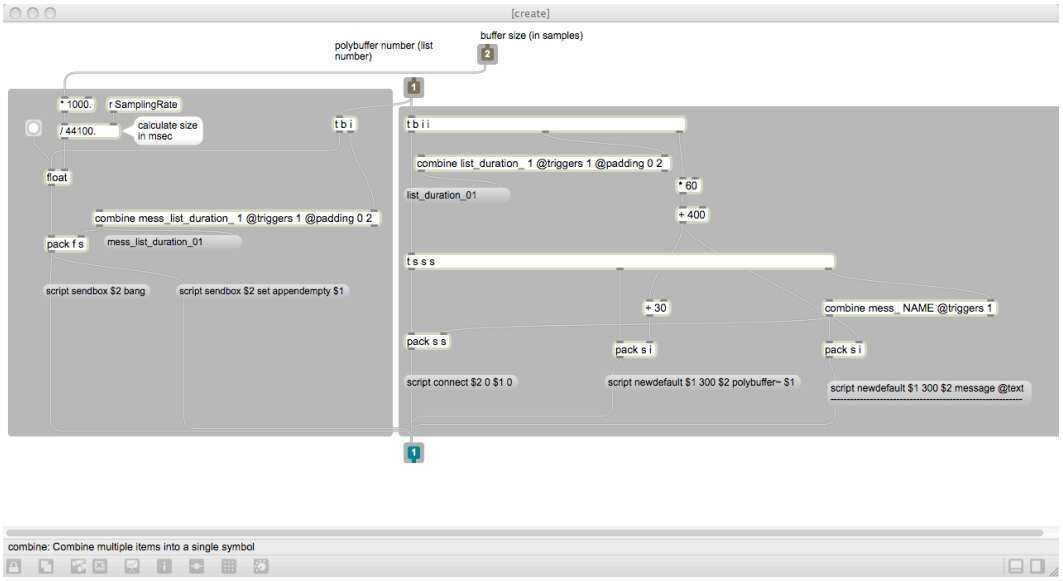

Figure 64: p create duration-list buffer

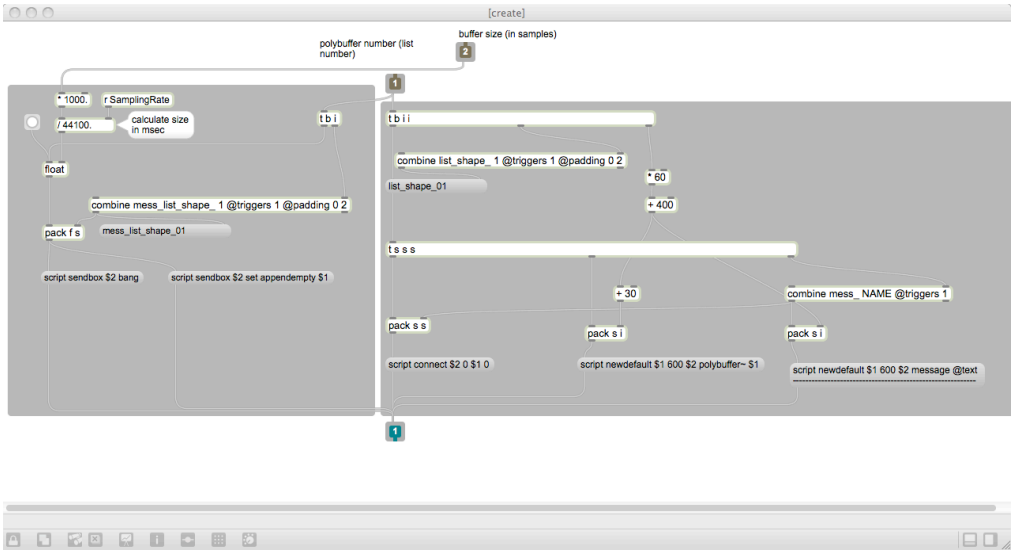

Figure 65: p create shape-list buffer

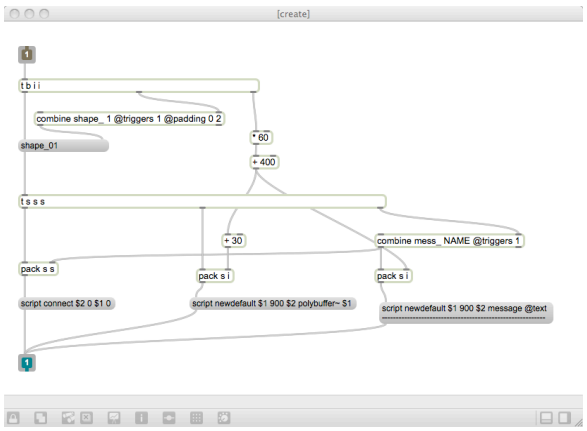

Figure 66: p create shape-list POLYbuffer

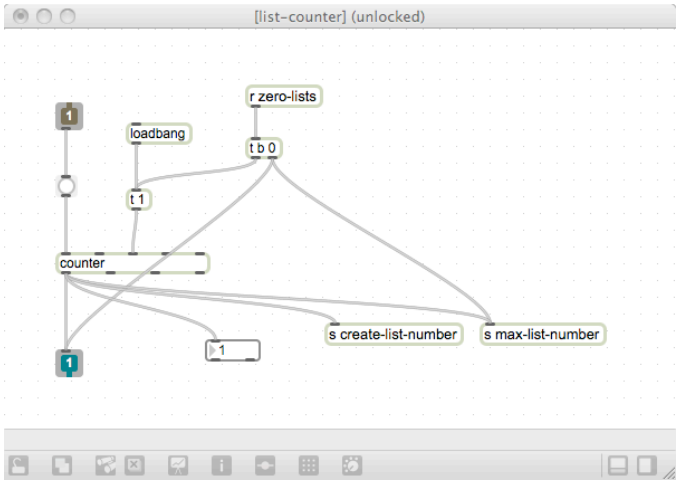

Figure 67: p list-counter

2) **DWSS_construction**

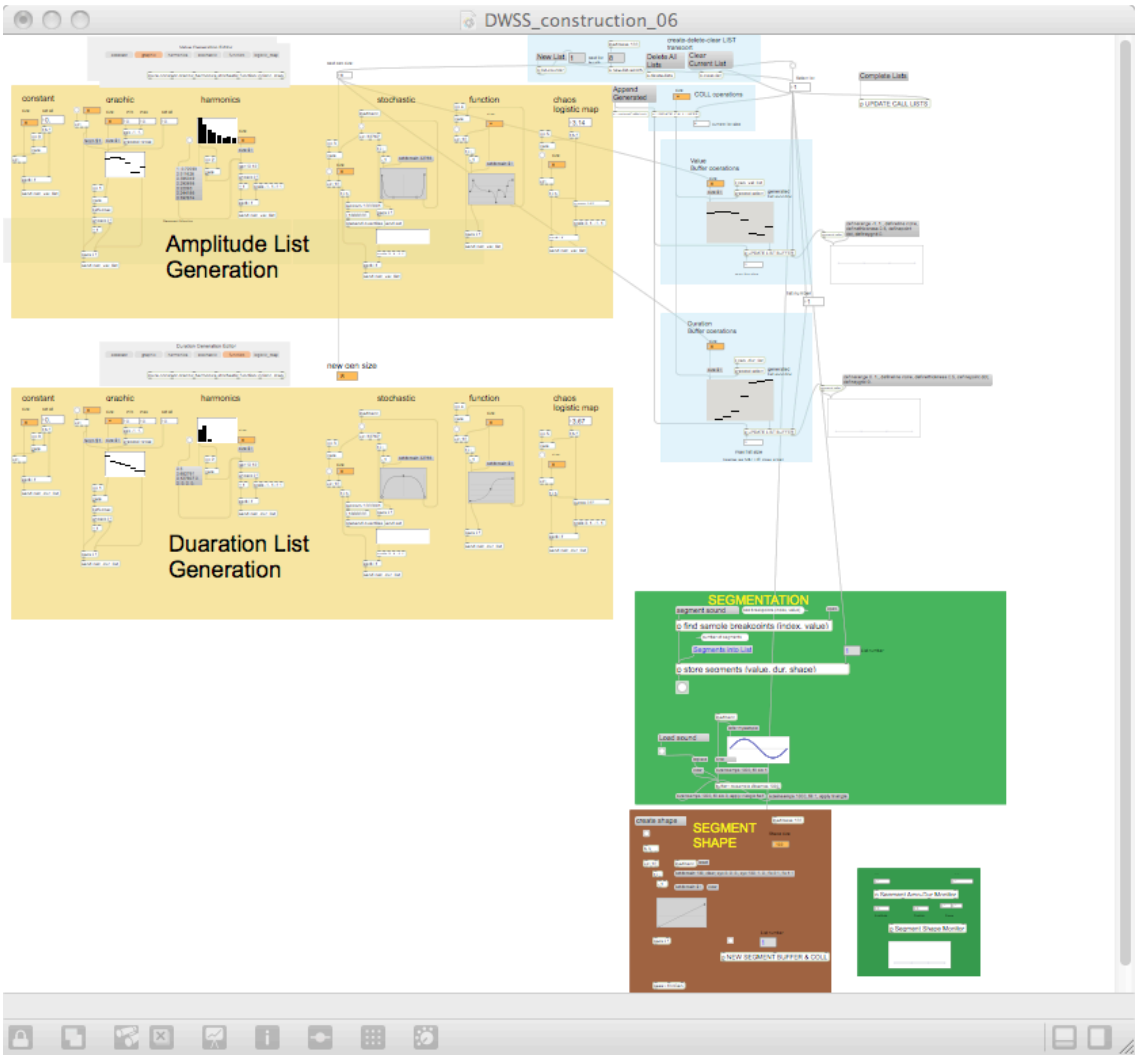

Figure 68: DWSS_construction

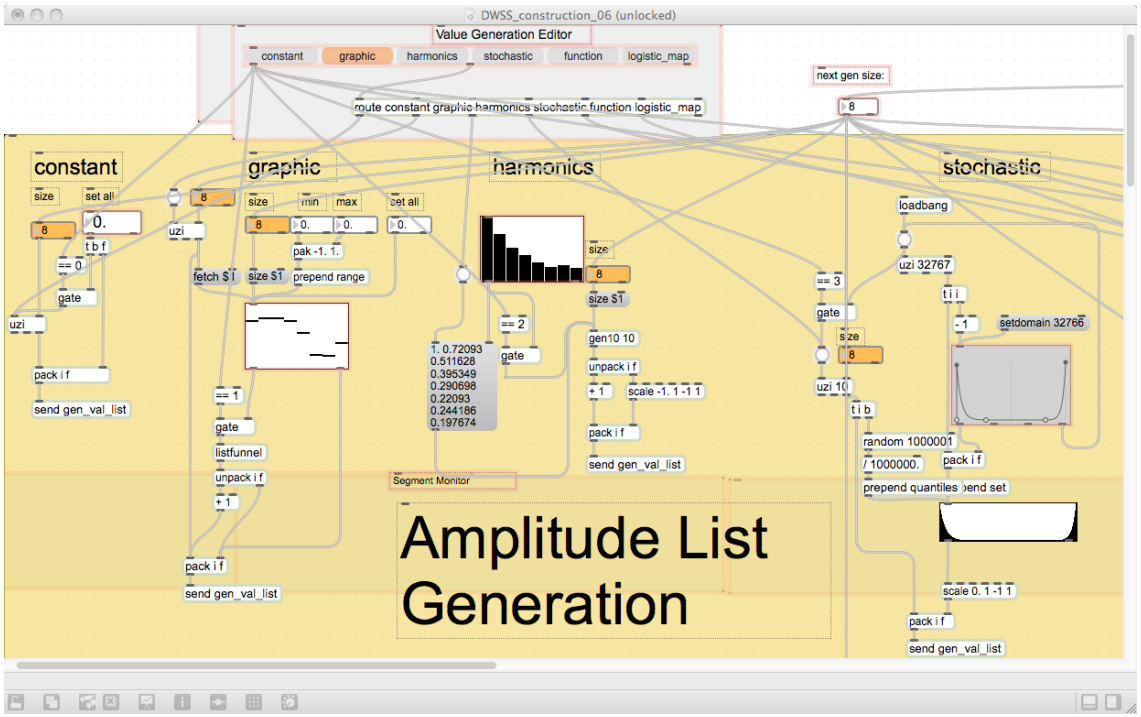

Figure 69: DWSS_construction (detail a)

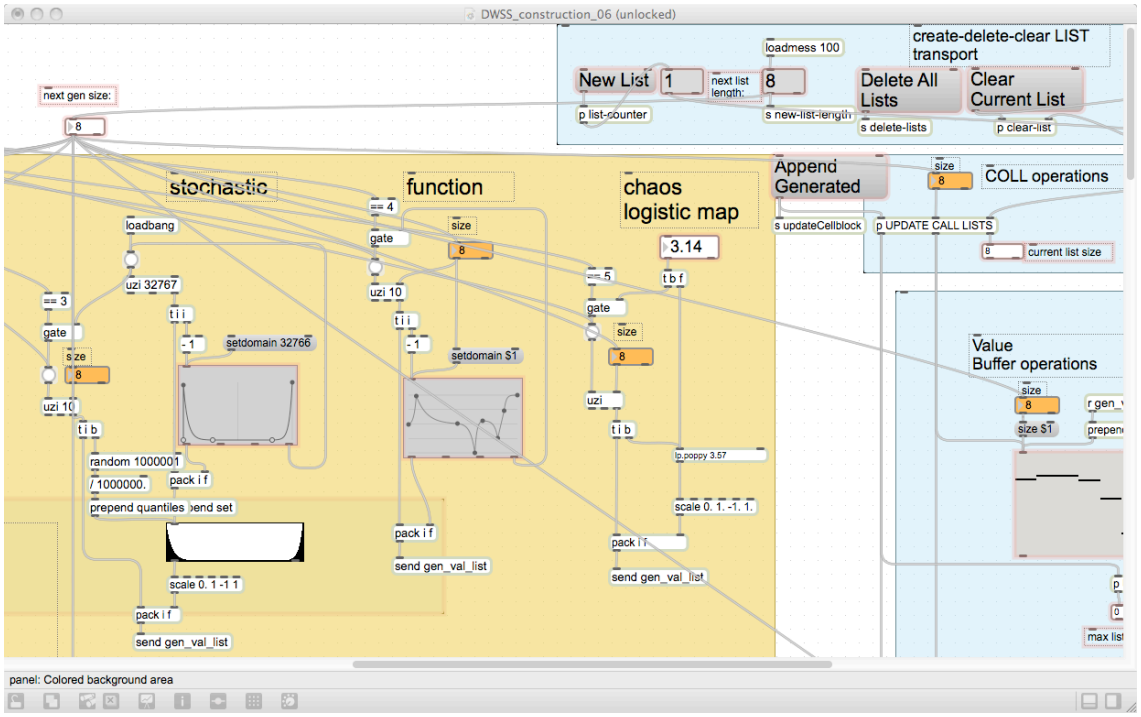

Figure 70: DWSS_construction (detail b)

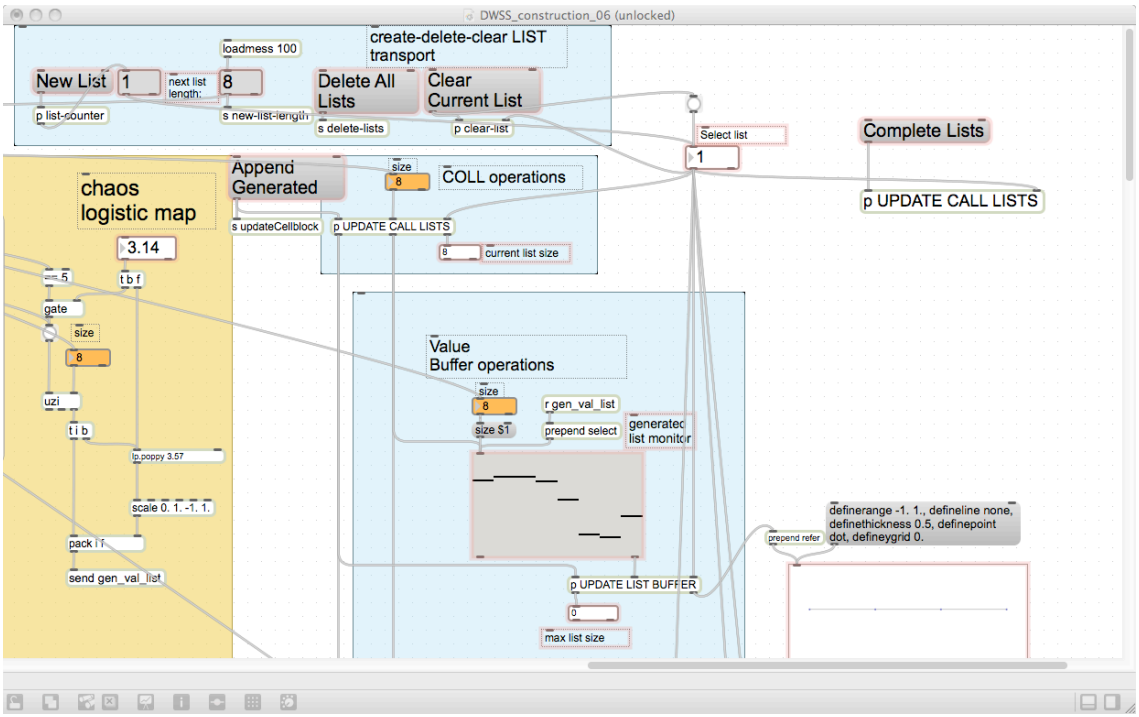

Figure 71: DWSS_construction (detail c)

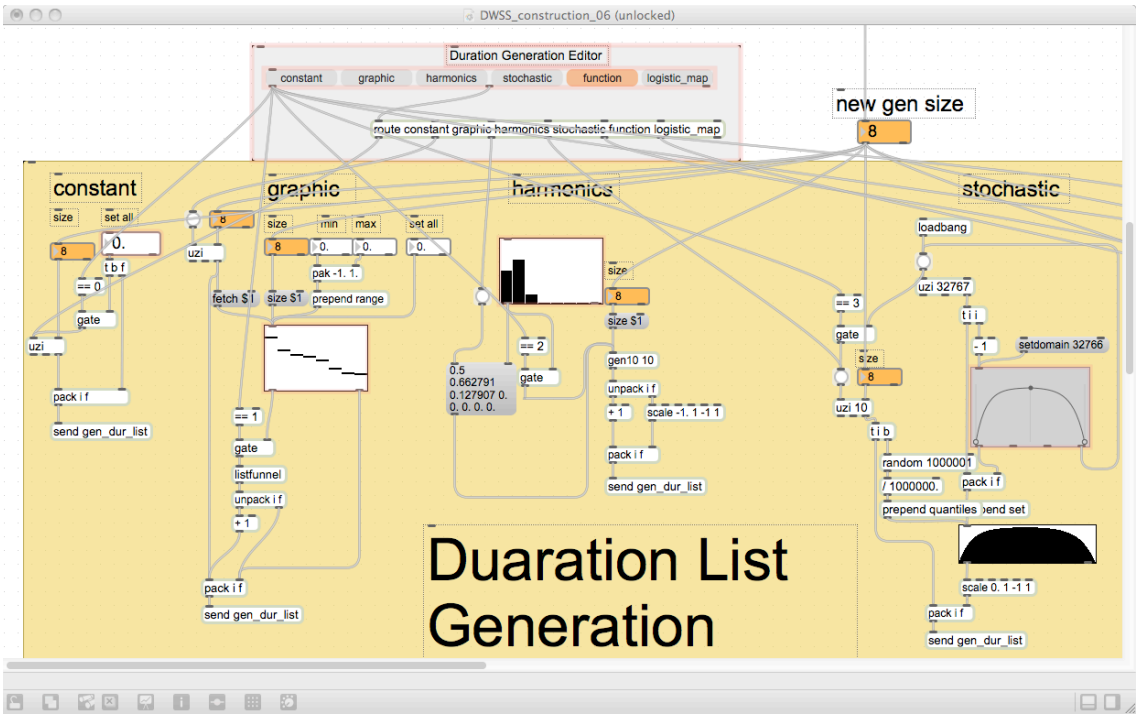

Figure 72: DWSS_construction (detail d)

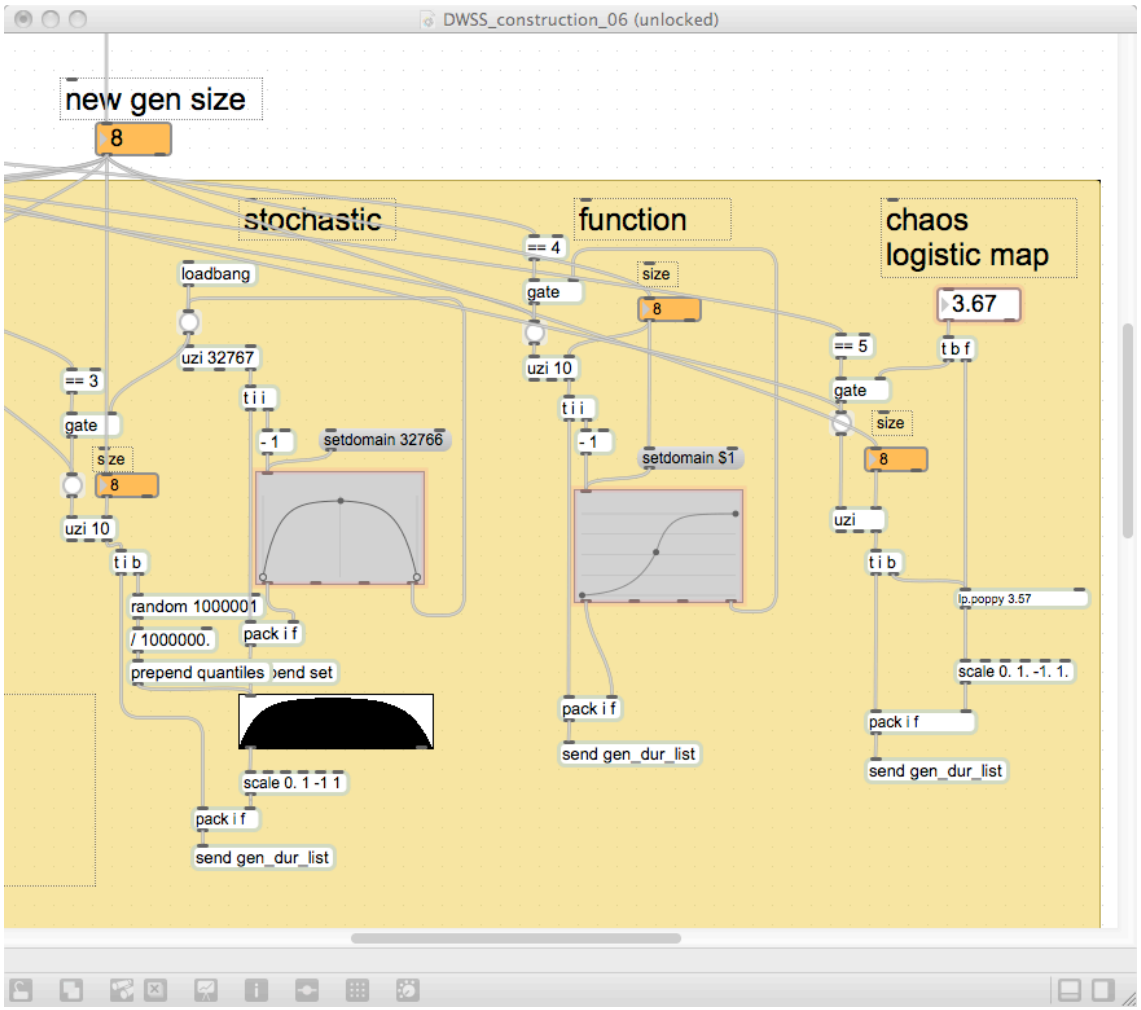

Figure 73: DWSS_construction (detail e)

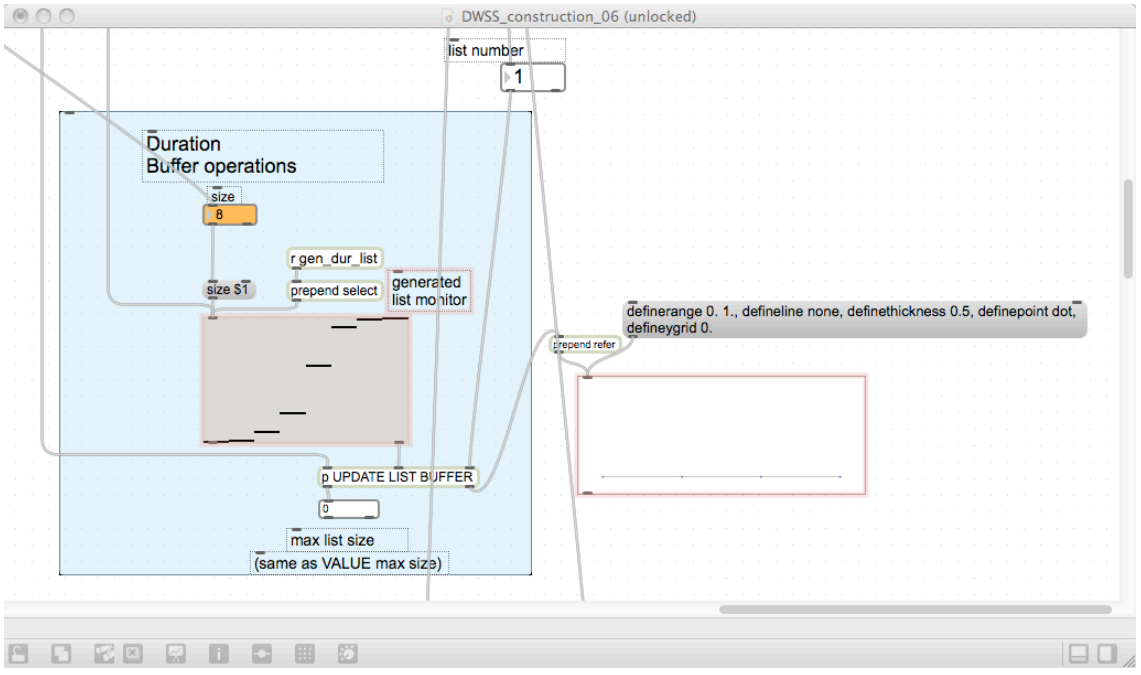

Figure 74: DWSS_construction (detail f)

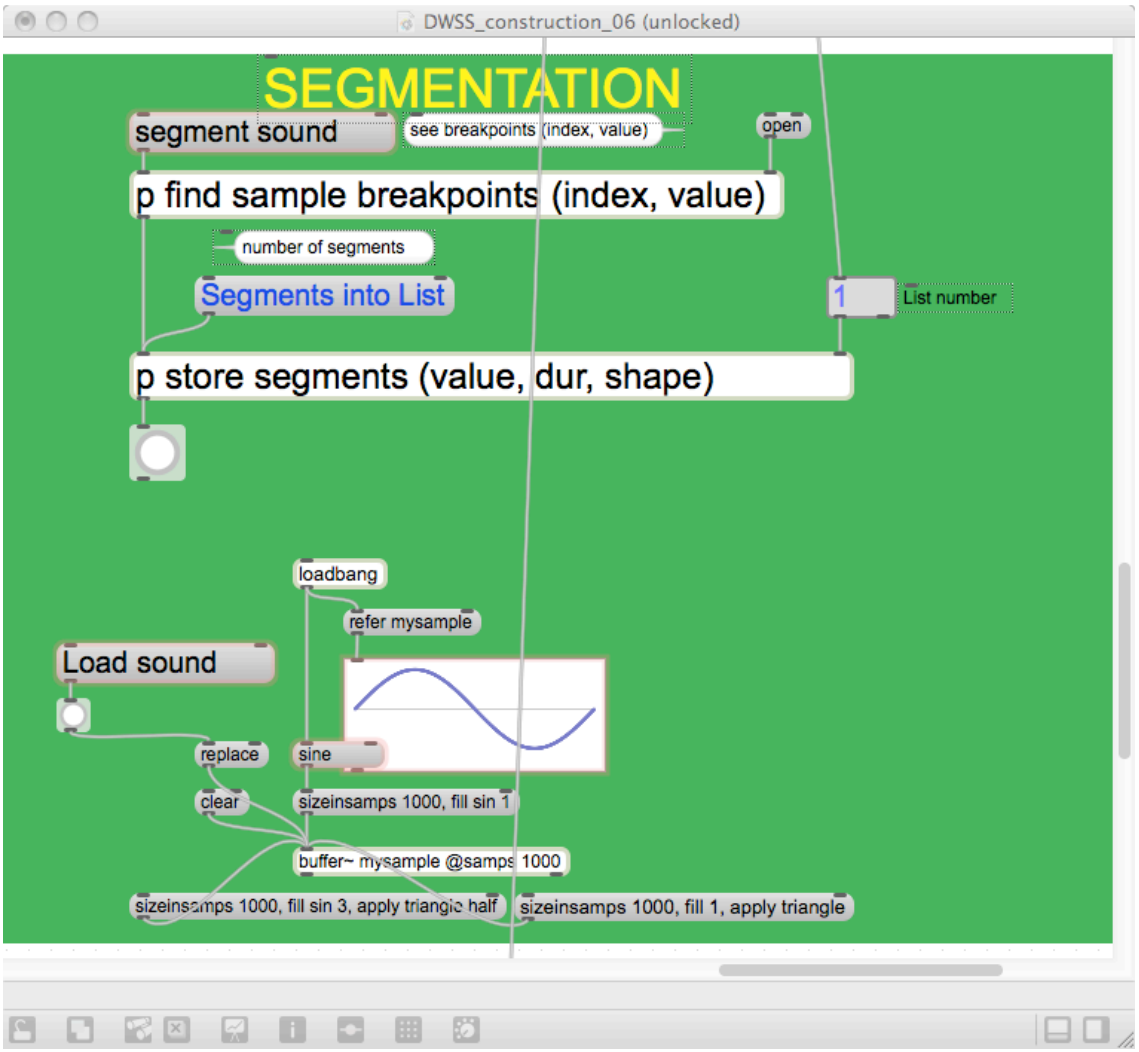

Figure 75: DWSS_construction (detail g)

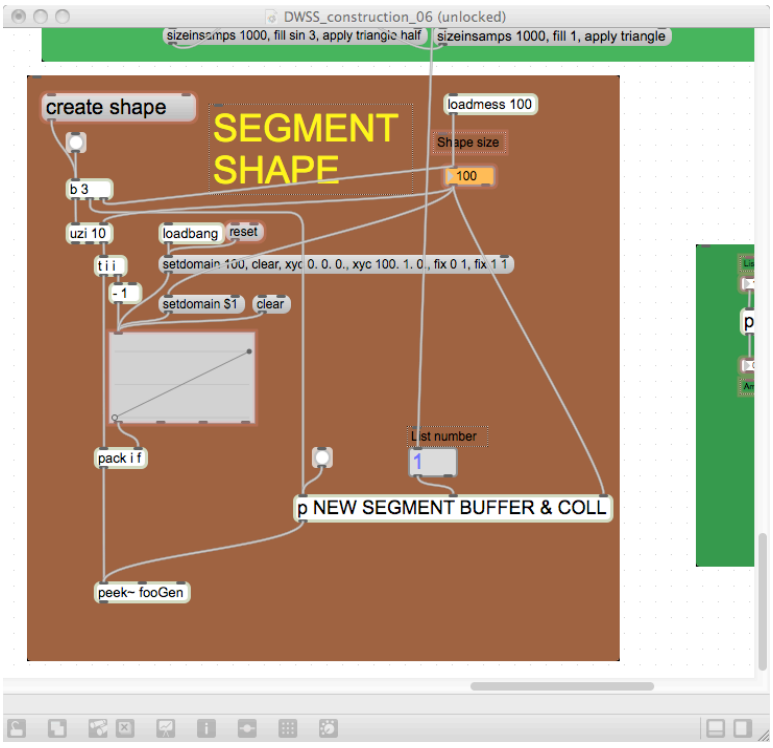

Figure 76: DWSS_construction (detail h)

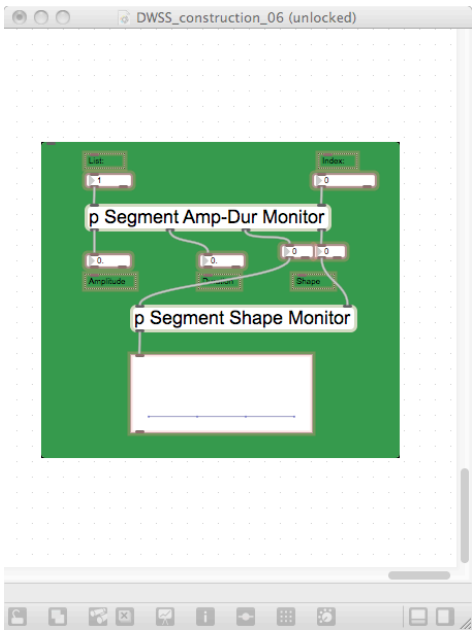

Figure 77: DWSS_construction (detail

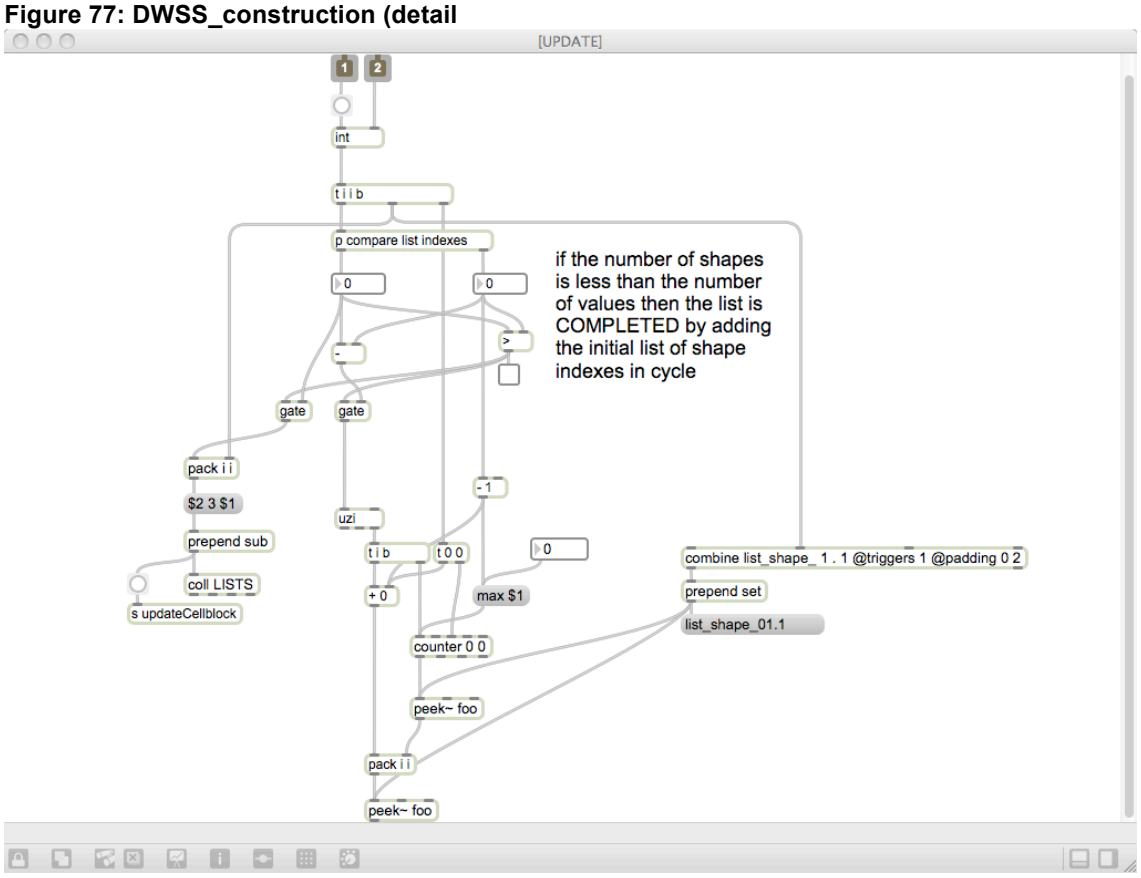

Figure 78: p UPDATE CALL LISTS (complete)

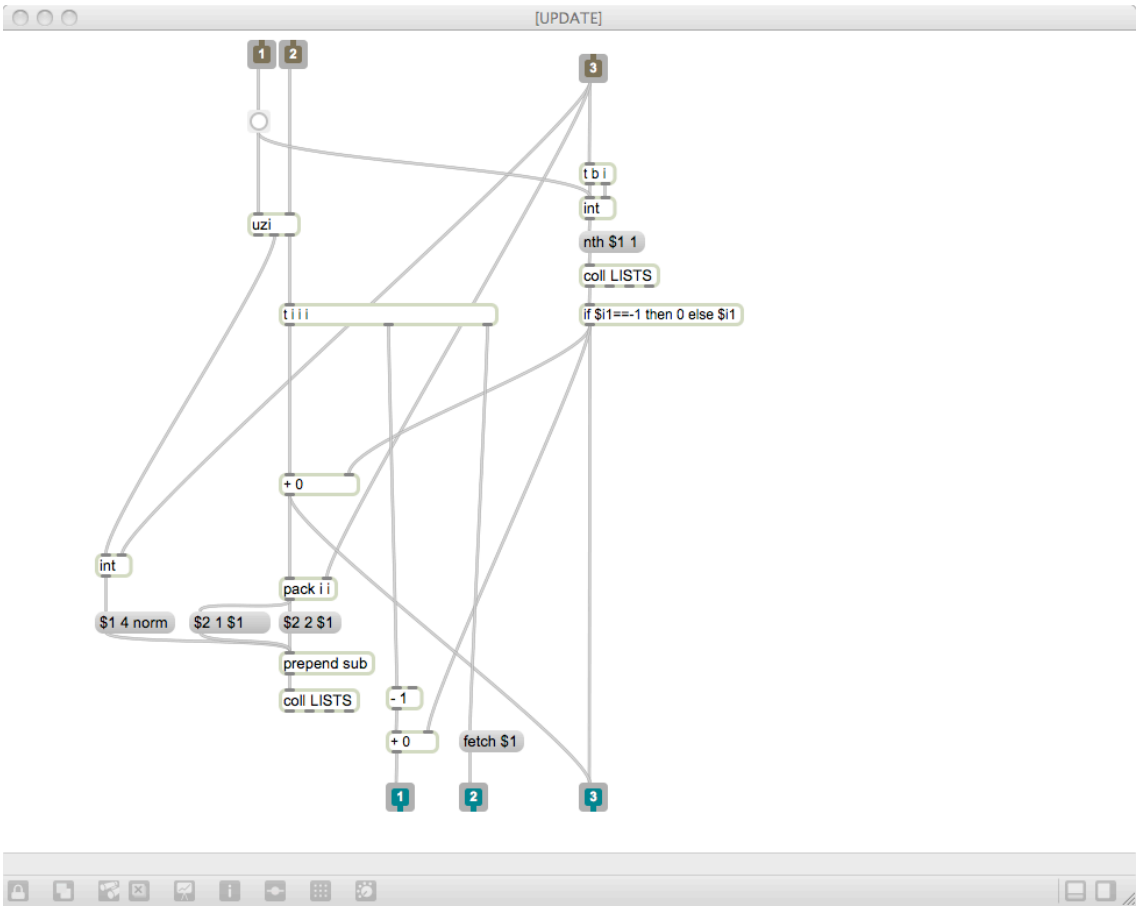

Figure 79: p UPDATE CALL LISTS (append)

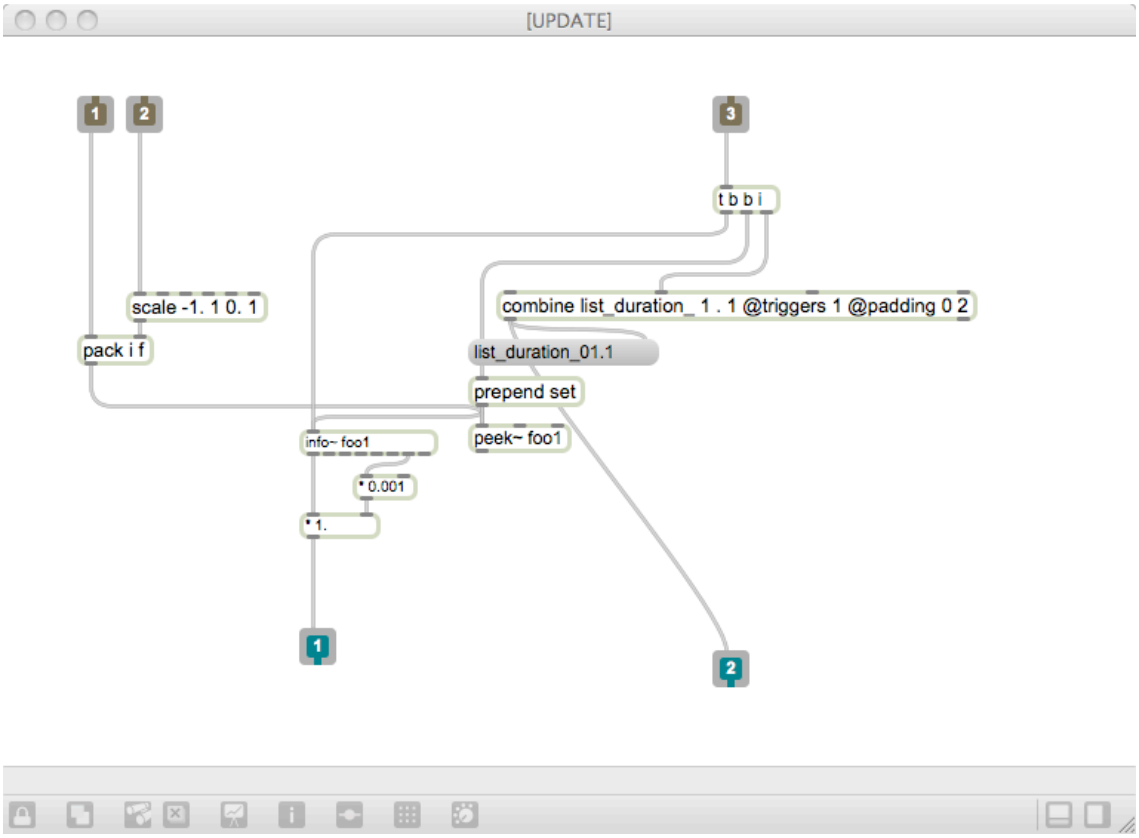

Figure 80: p UPDATE LIST BUFFER

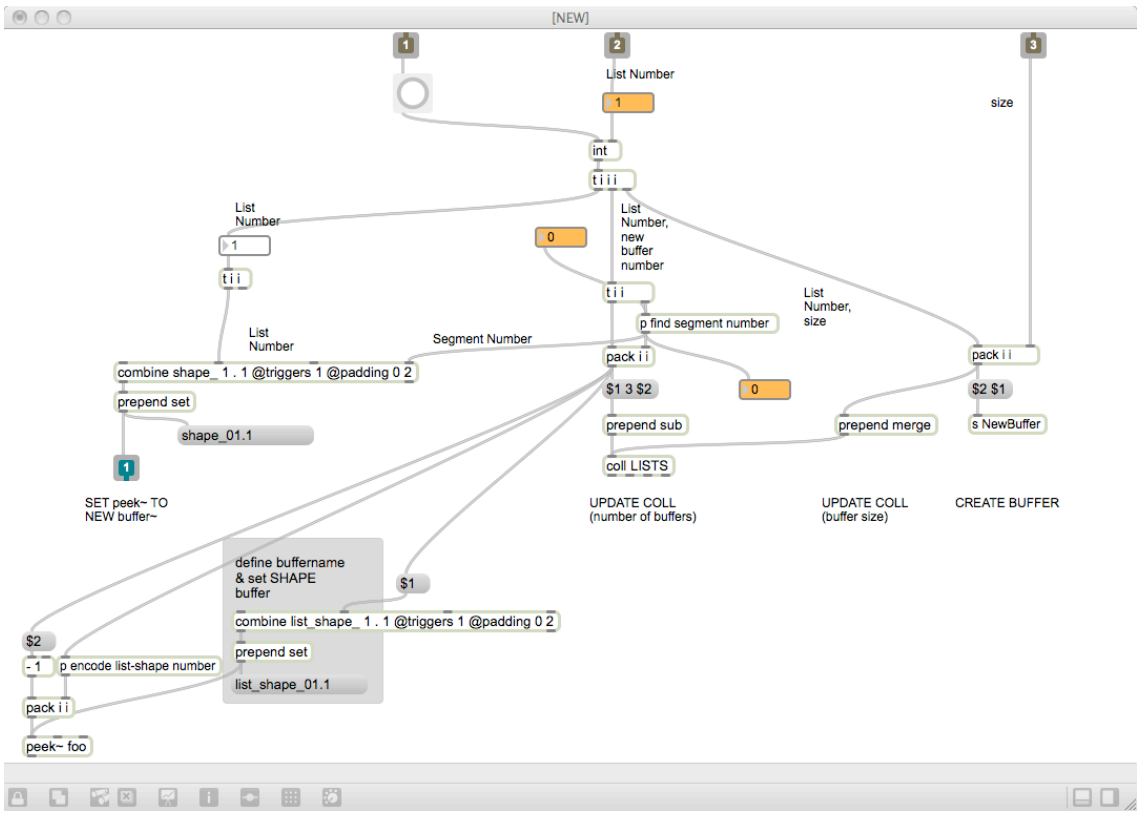

Figure 81: p NEW SEGMENT BUFFER & COLL

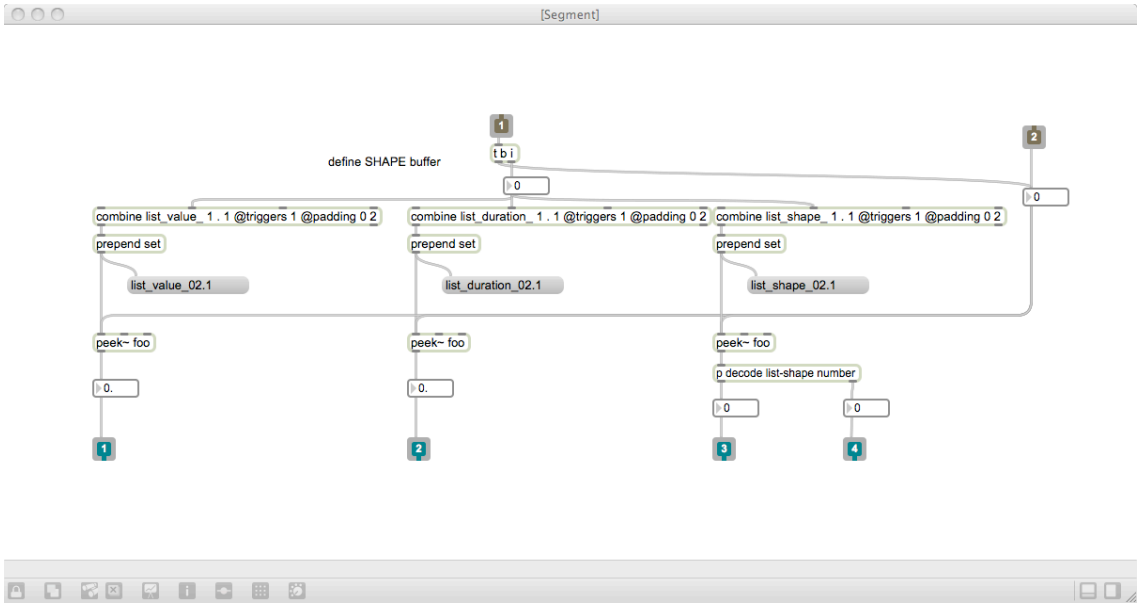

Figure 82: p Segment Amp-Dur Monitor

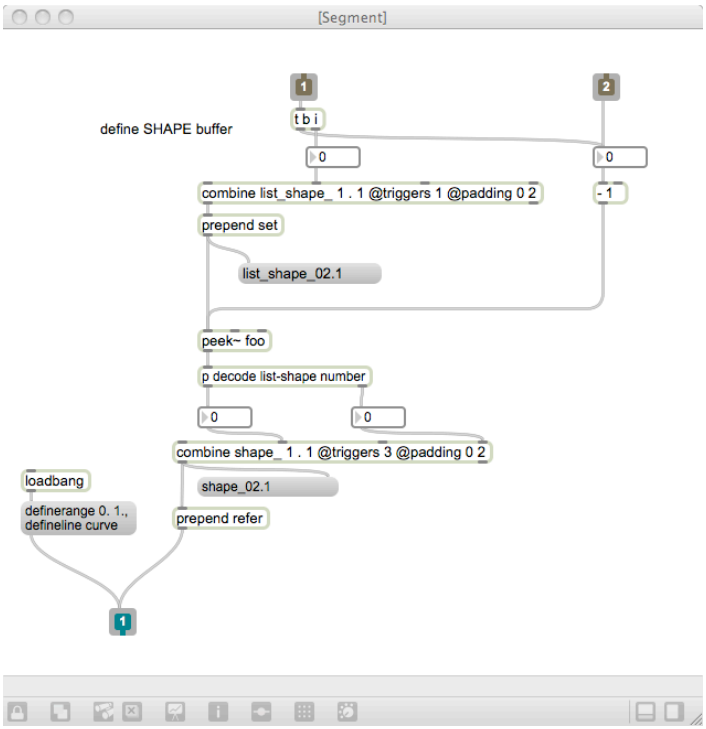

Figure 83: p Segment Shape Monitor

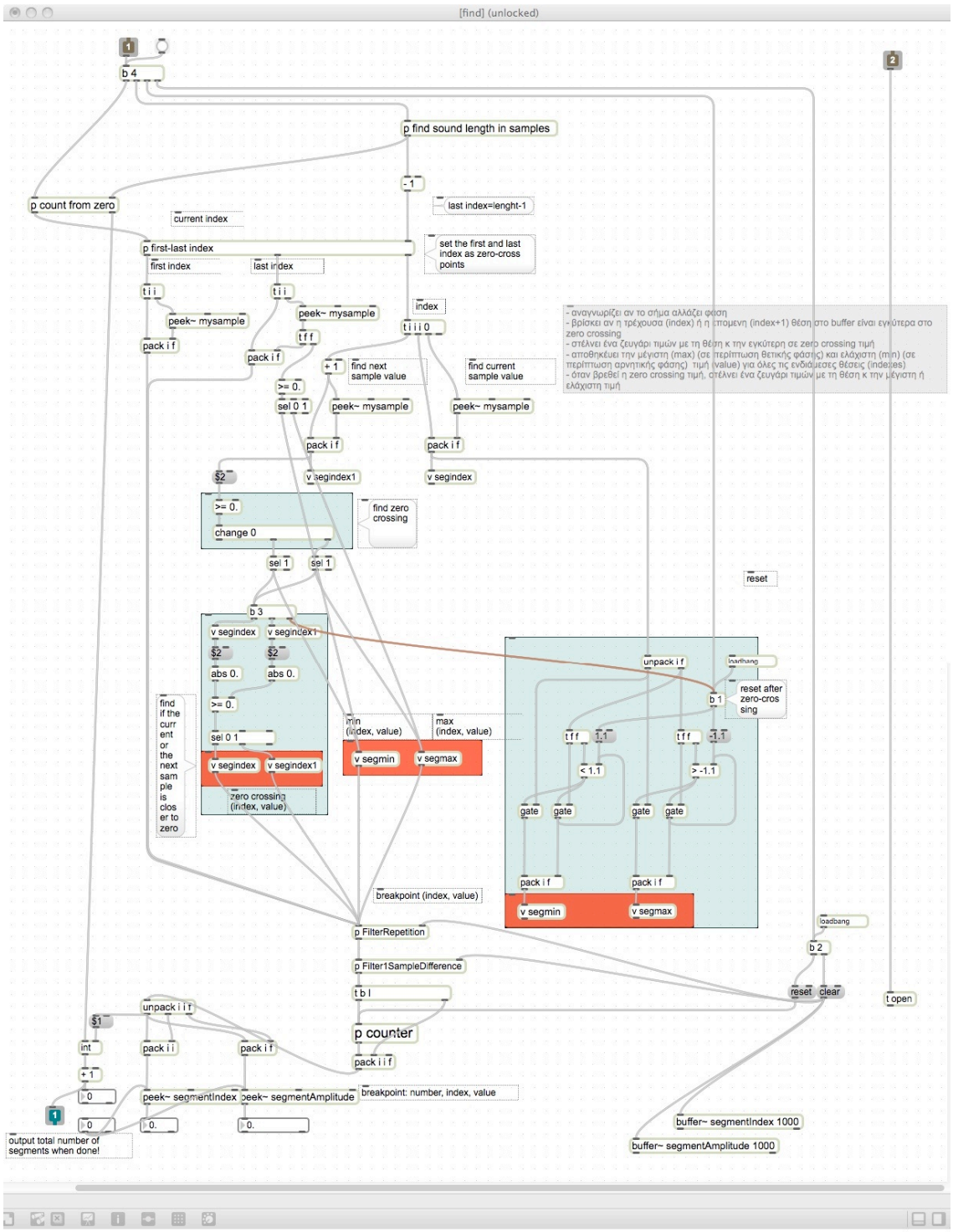

Figure 84: p find sample breakpoints (index, value)

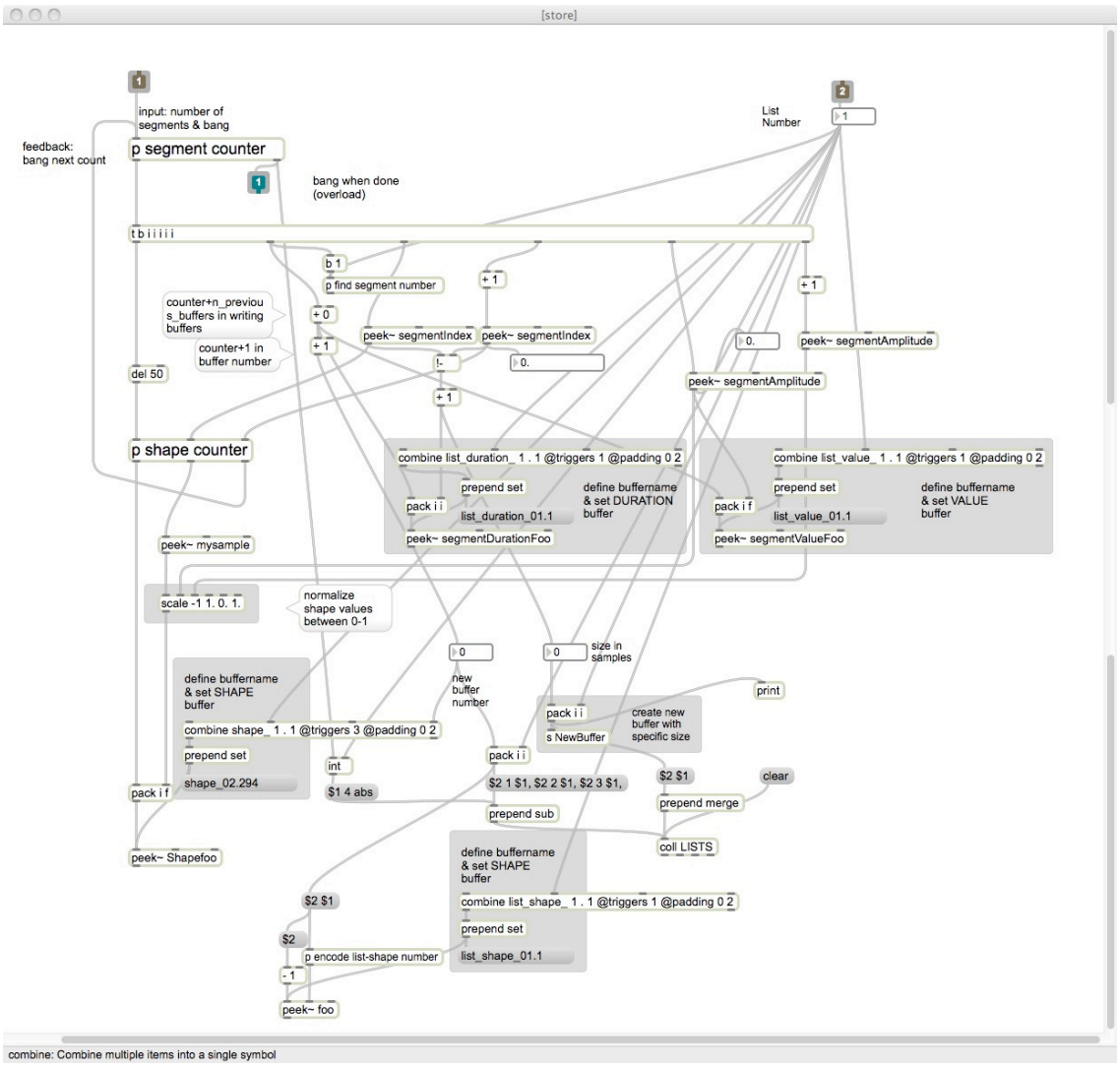

Figure 85: p store segments (value, dur, shape)

3) DWSS_transformation

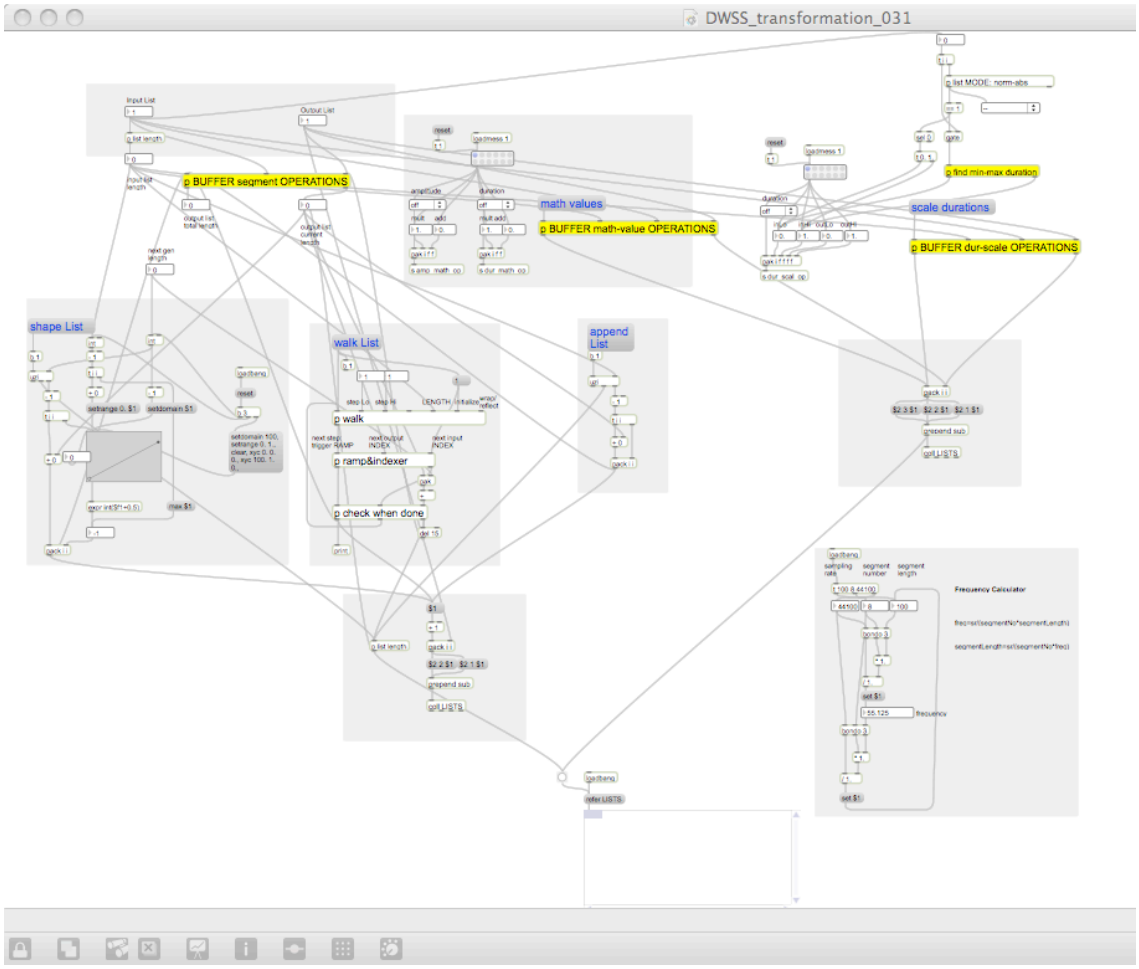

Figure 86: DWSS_transformation

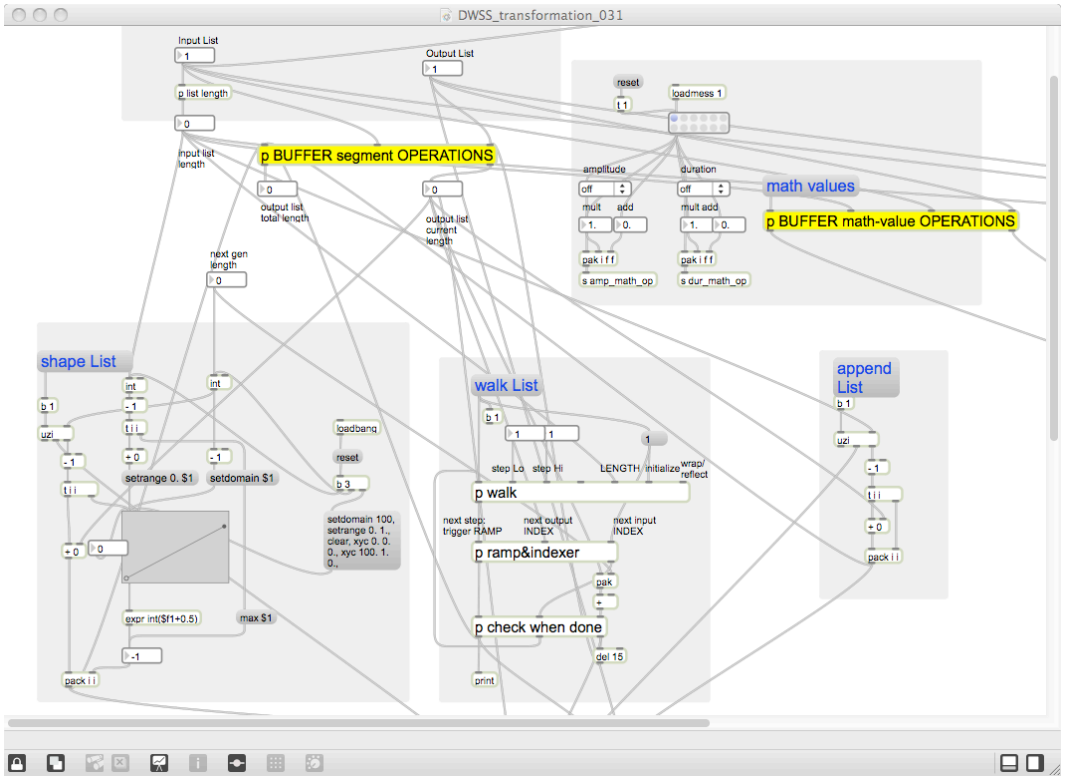

Figure 87: DWSS_transformation (detail a)

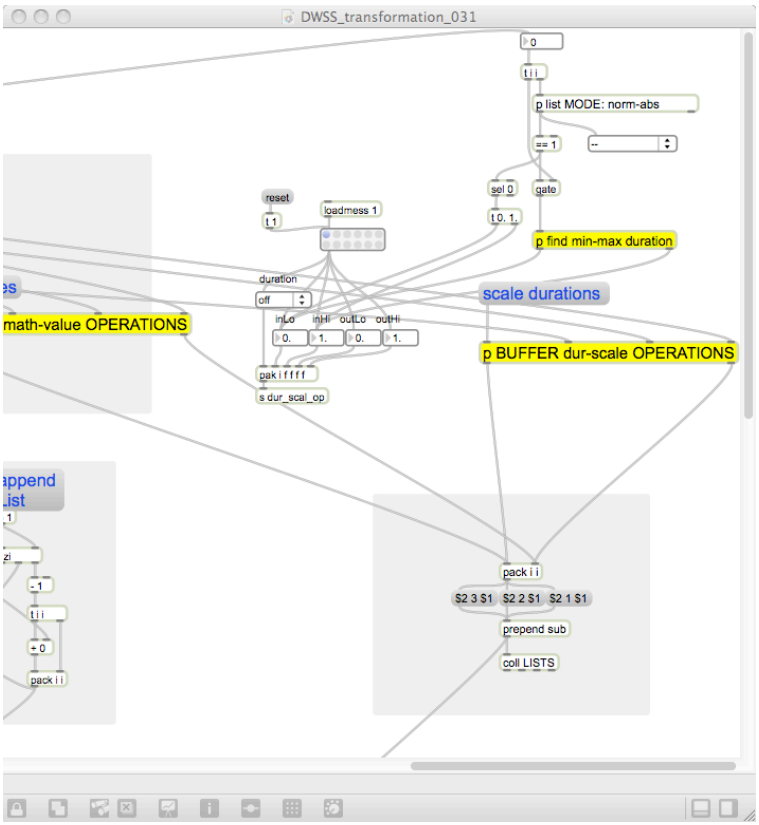

Figure 88: DWSS_transformation (detail b)

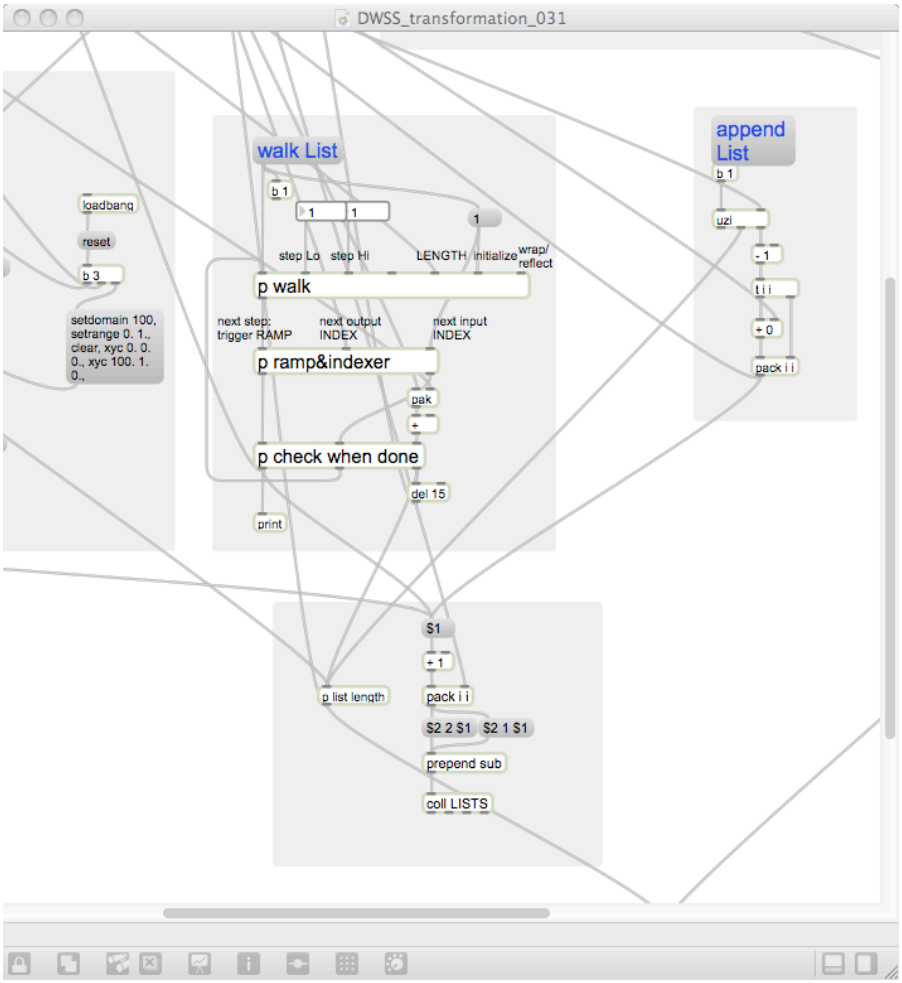

Figure 89: DWSS_transformation (detail c)

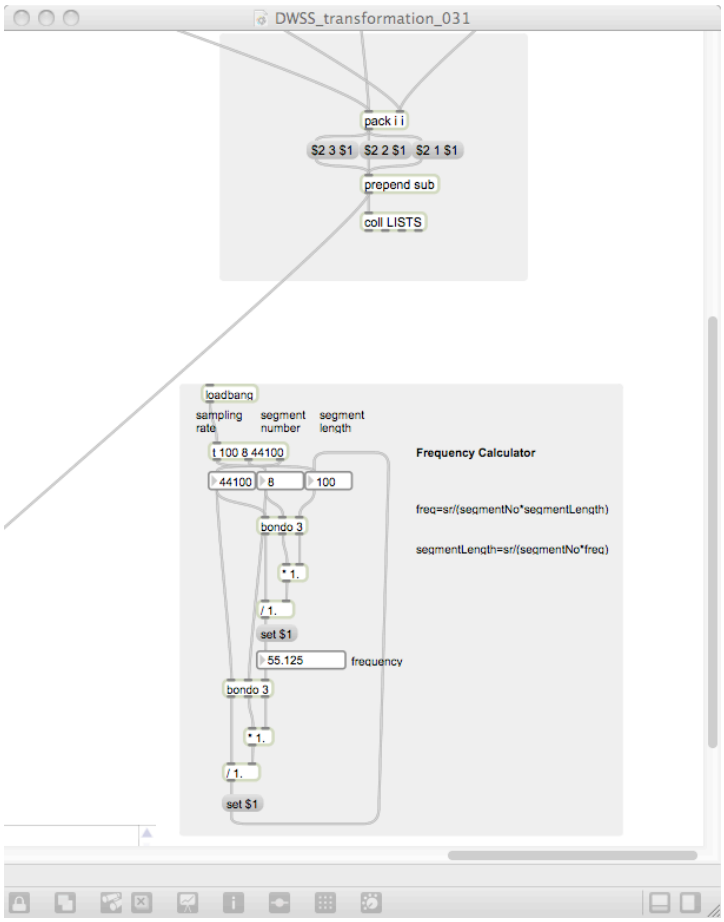

Figure 90: DWSS_transformation (detail d)

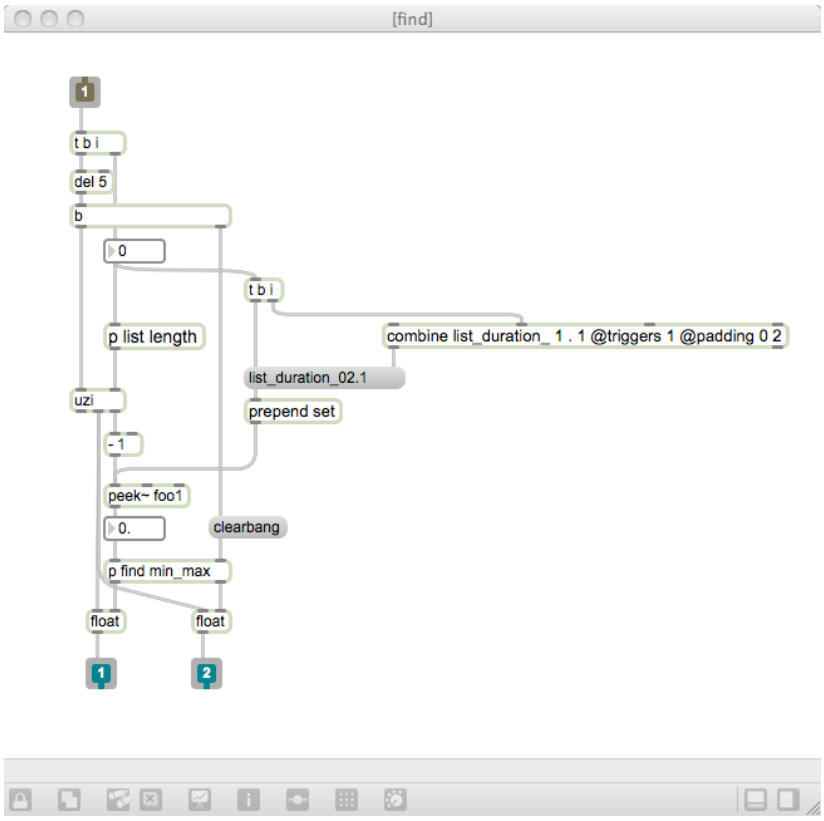

Figure 91: p find min-max duration

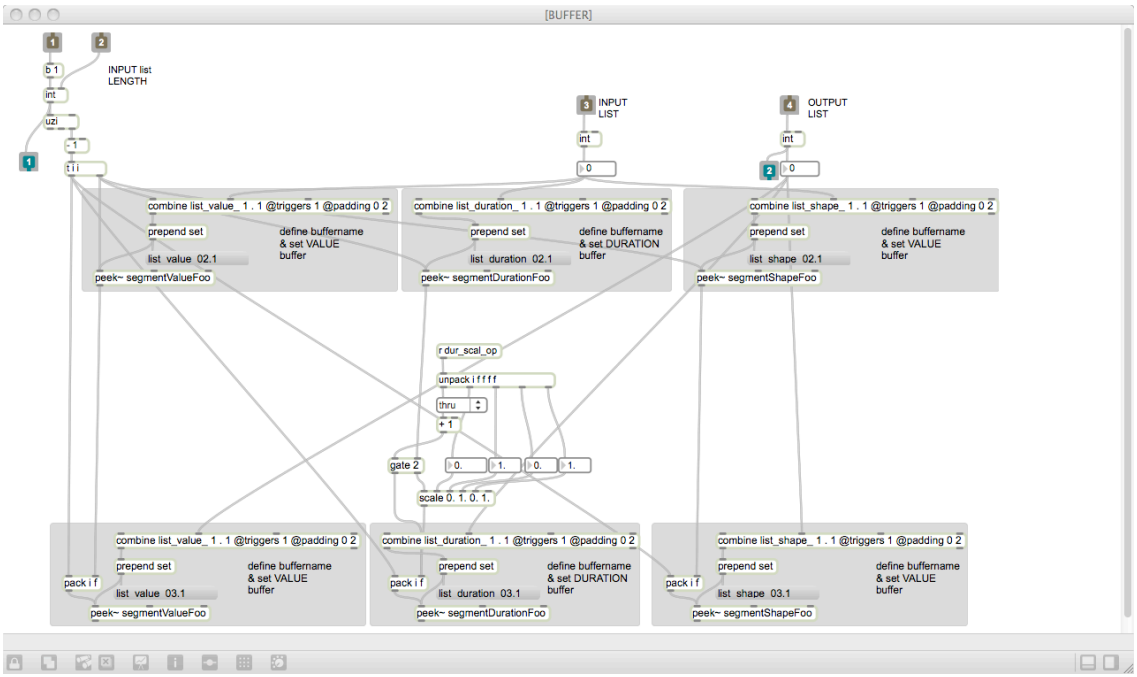

Figure 92: p BUFFER segment OPERATIONS

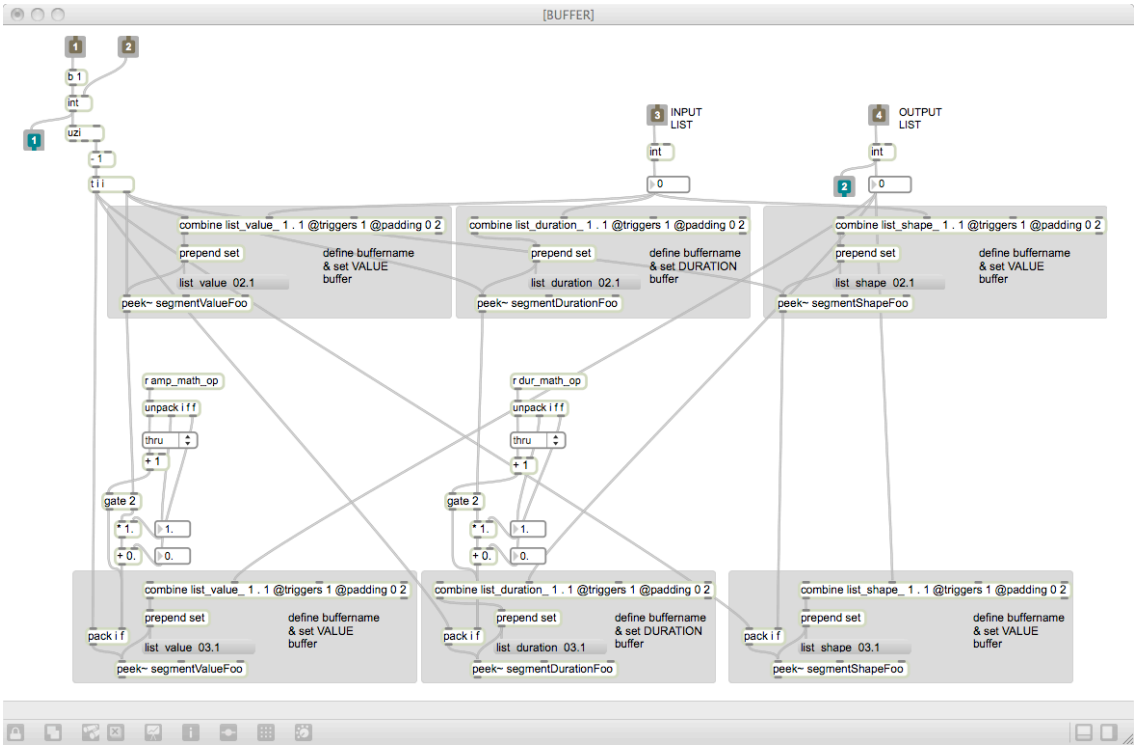

Figure 93: p BUFFER math-value OPERATIONS

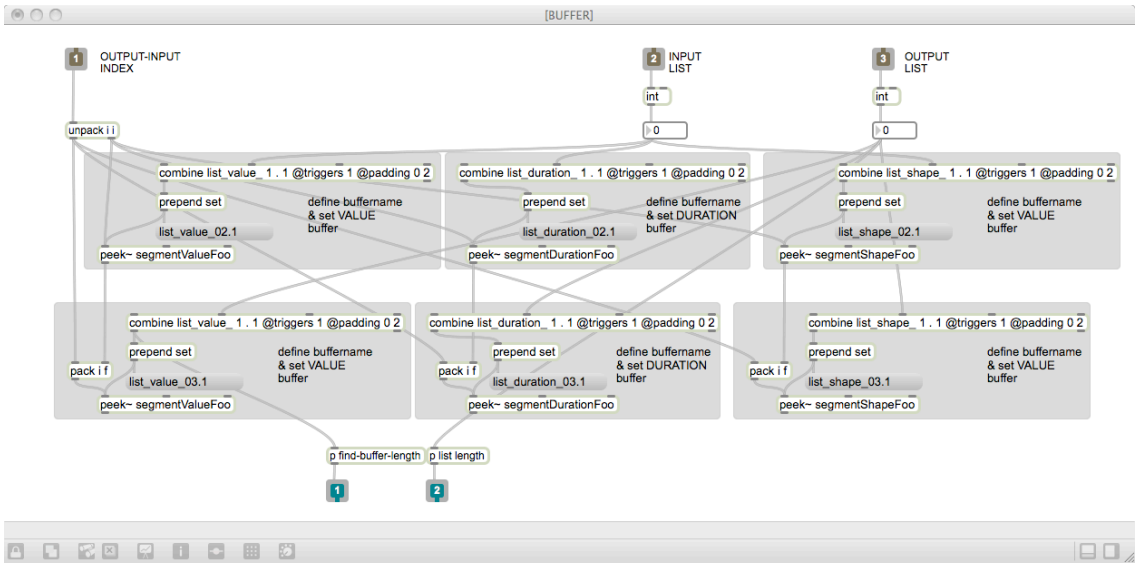

Figure 94: p BUFFER dur-scale OPERATIONS

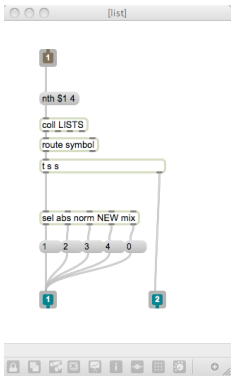

Figure 95: p list MODE: norm-abs

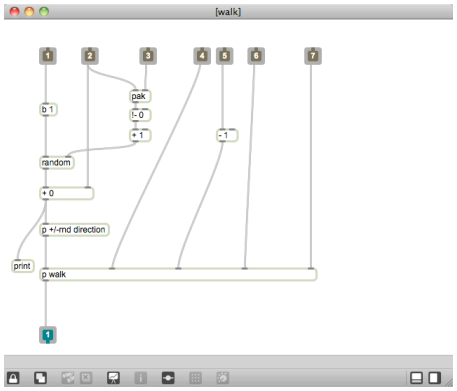

Figure 96: p walk

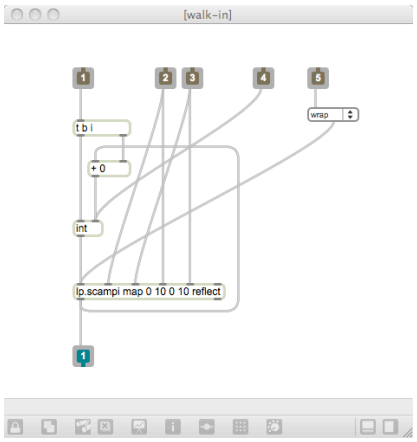

Figure 97: p walk-in

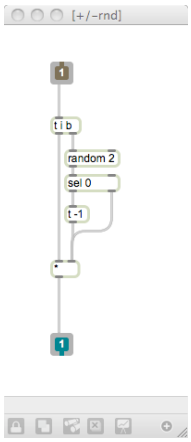

Figure 98: p +/-rnd direction

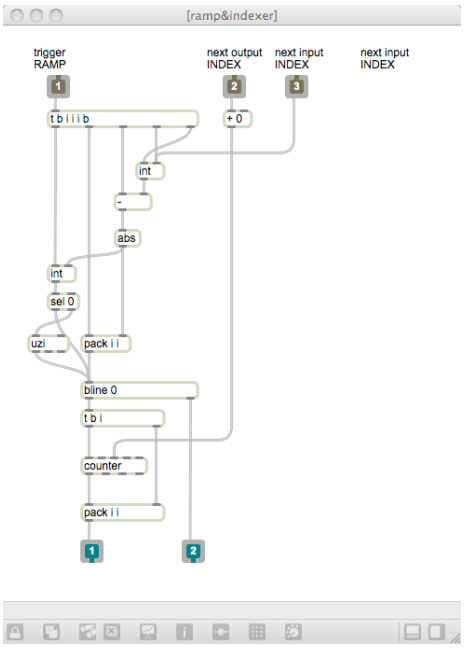

Figure 99: p ramp&indexer

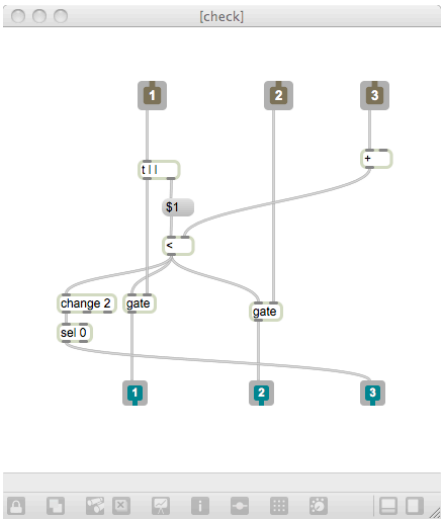

Figure 100: p check when done

4) DWSS_group

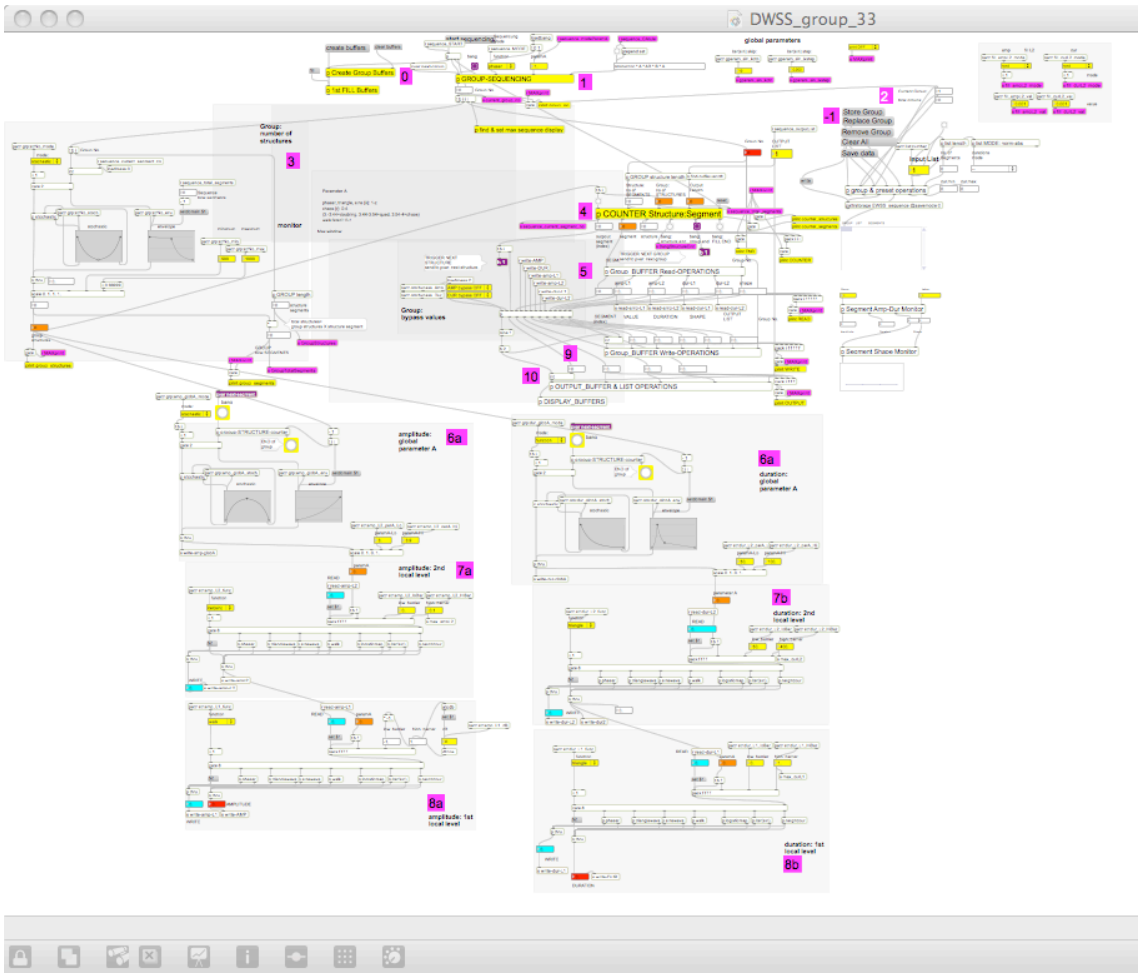

Figure 101: DWSS_group

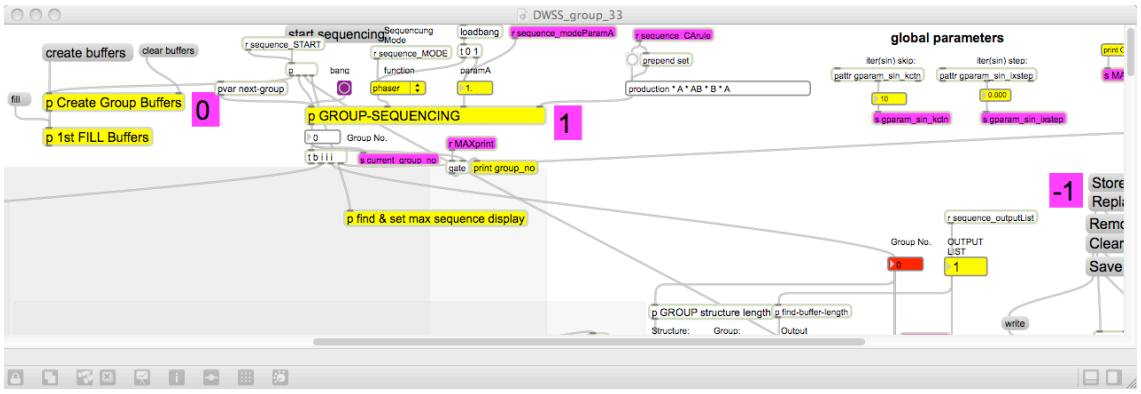

Figure 102: DWSS_group (detail a)

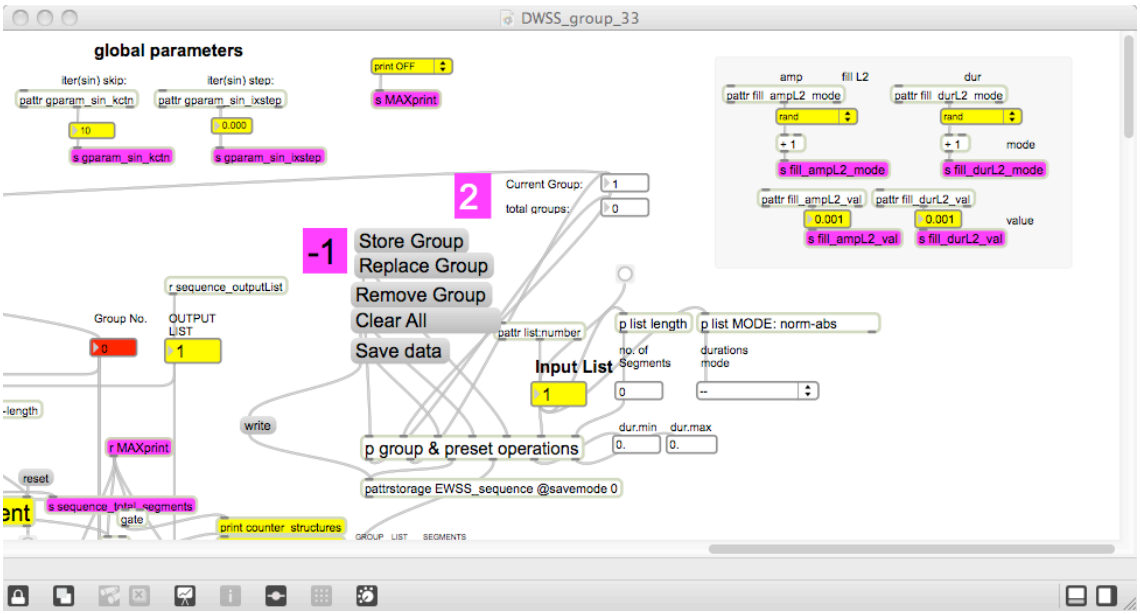

Figure 103: DWSS_group (detail b)

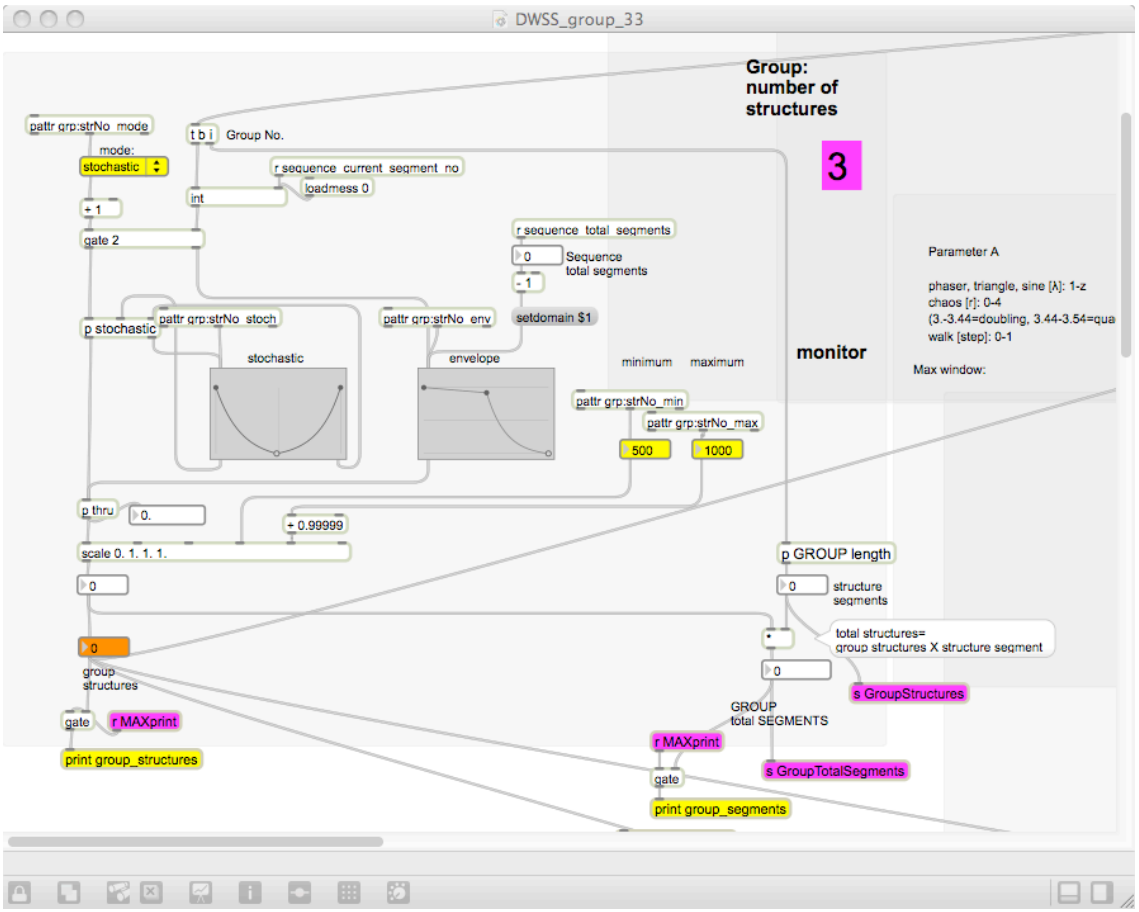

Figure 104: DWSS_group (detail c)

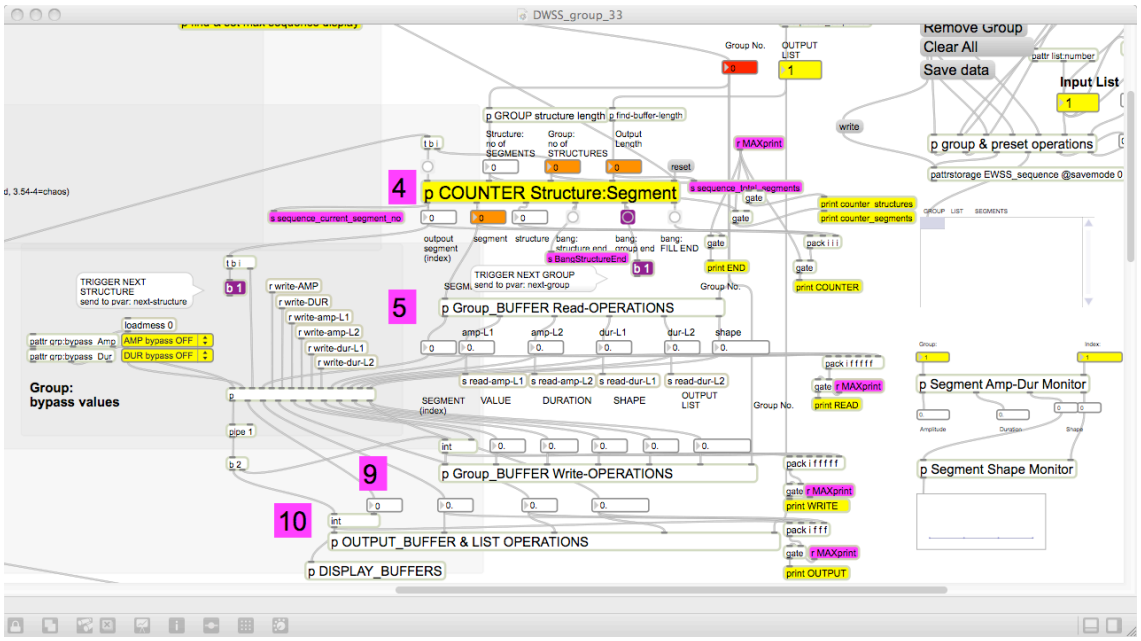

Figure 105: DWSS_group (detail d)

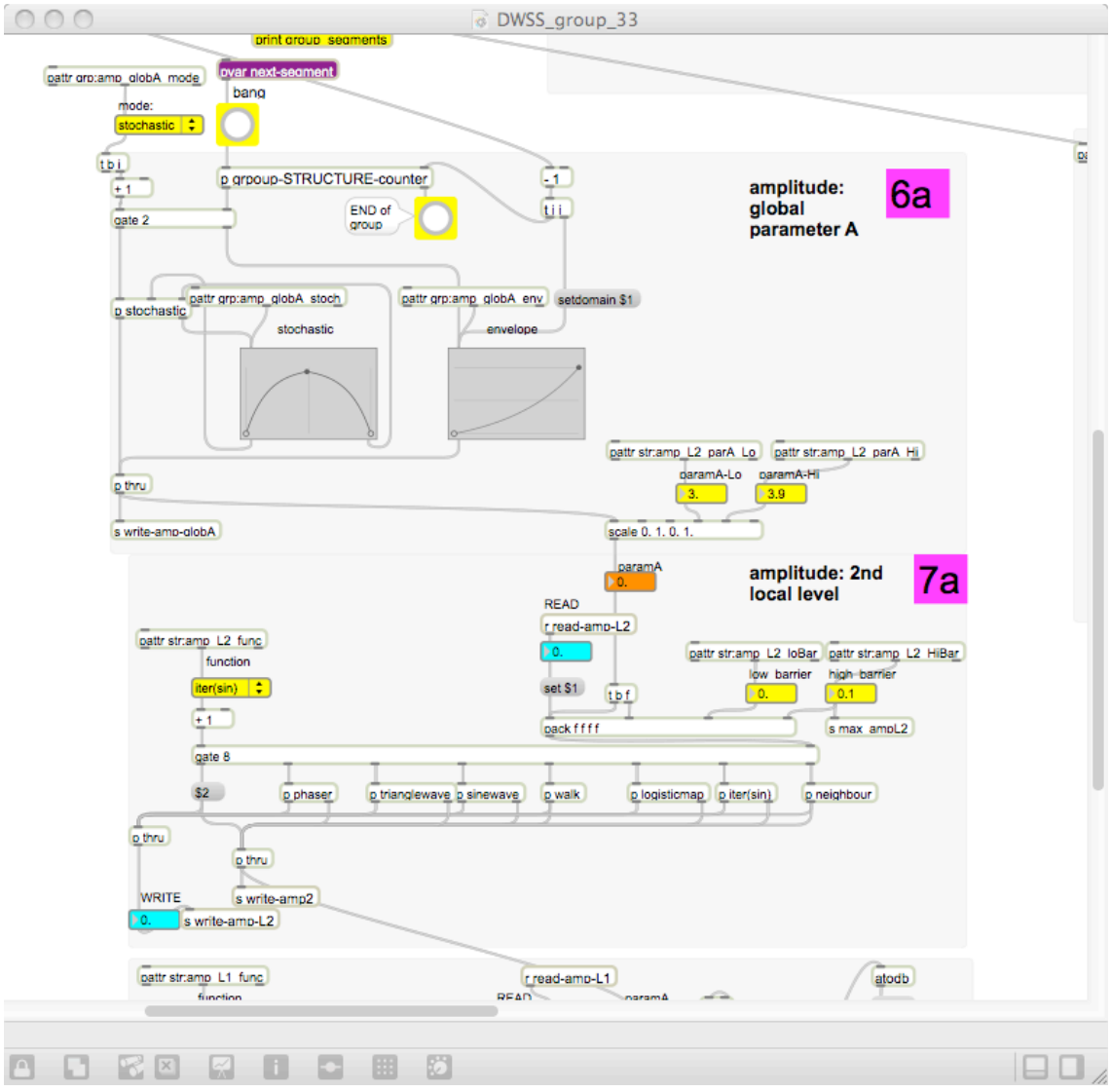

Figure 106: DWSS_group (detail e)

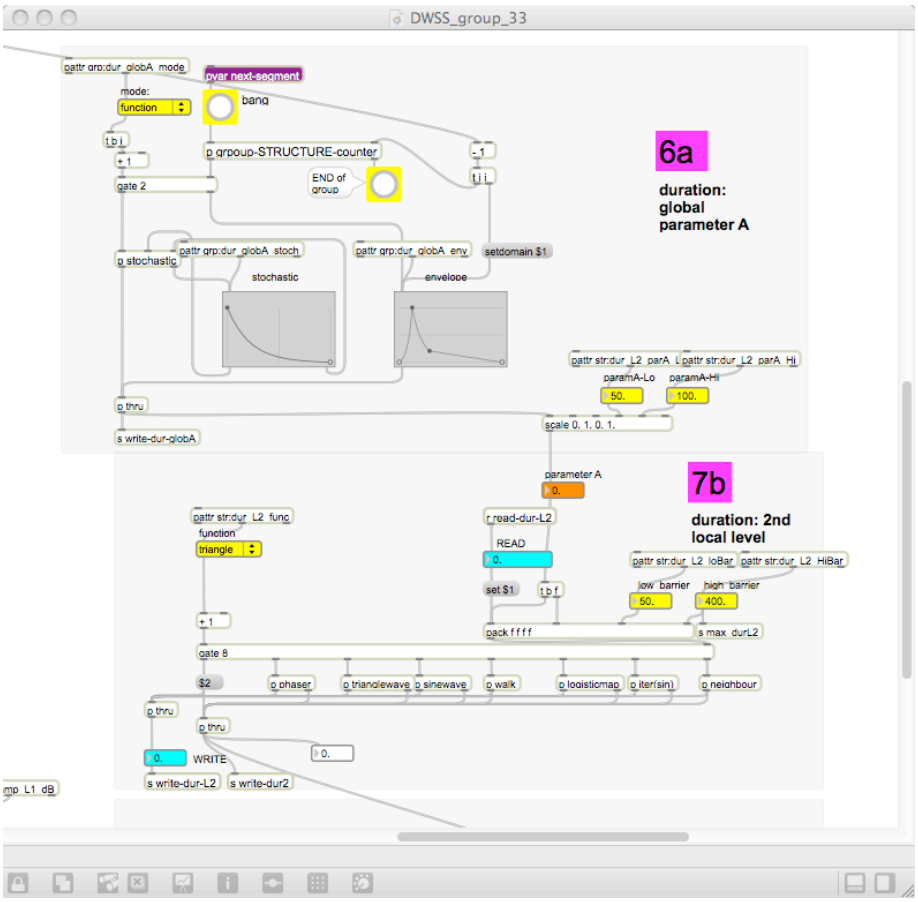

Figure 107: DWSS_group (detail f)

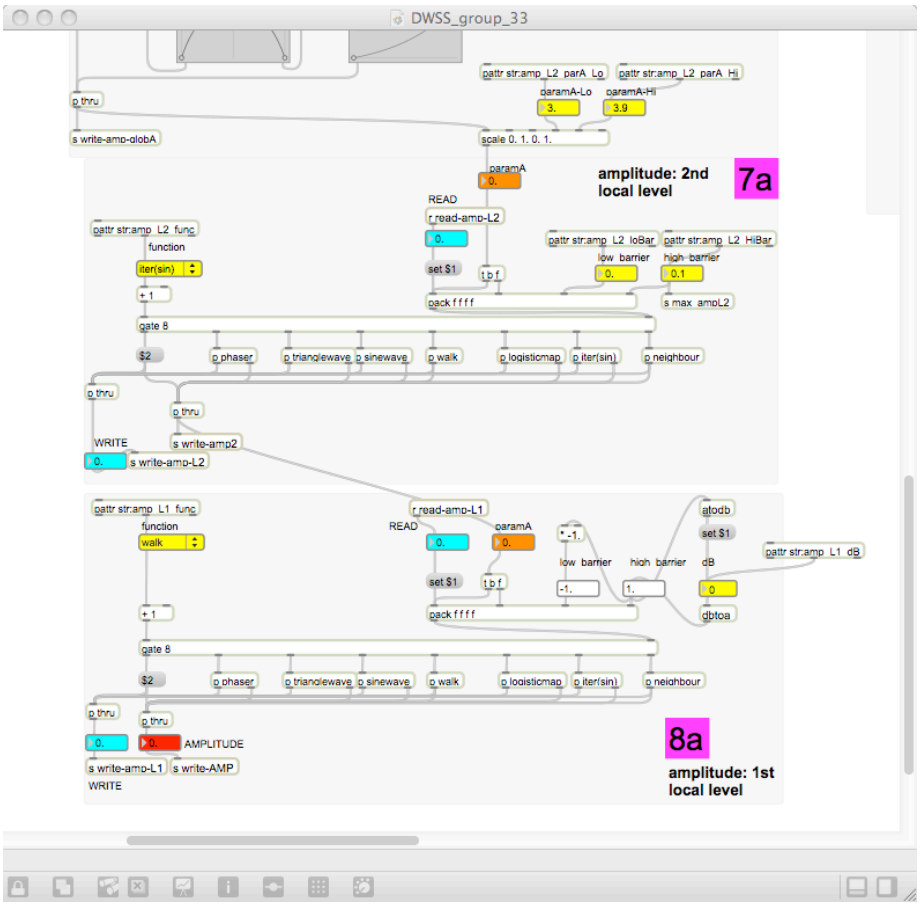

Figure 108: DWSS_group (detail g)

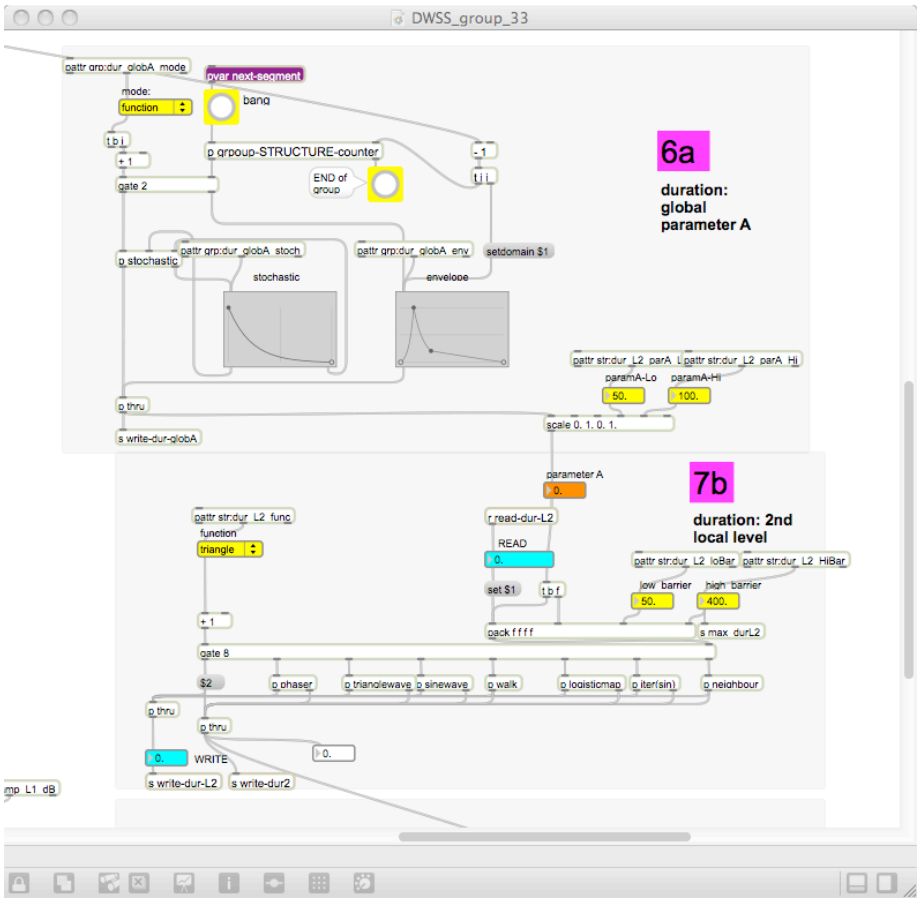

Figure 109: DWSS_group (detail h)

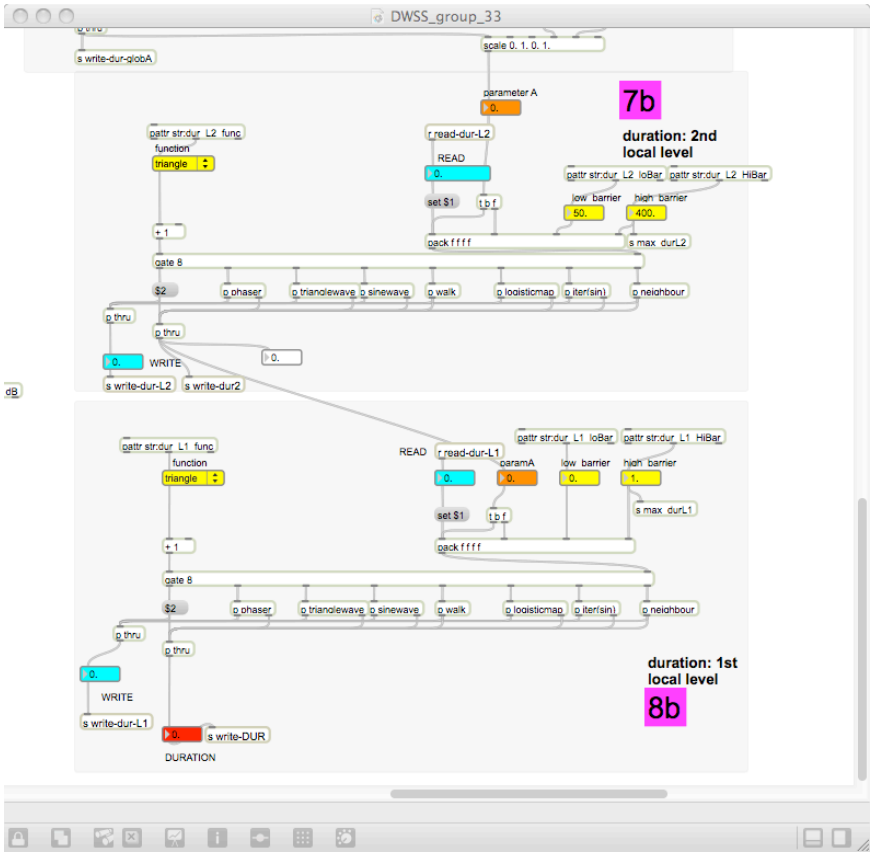

Figure 110: DWSS_group (detail i)

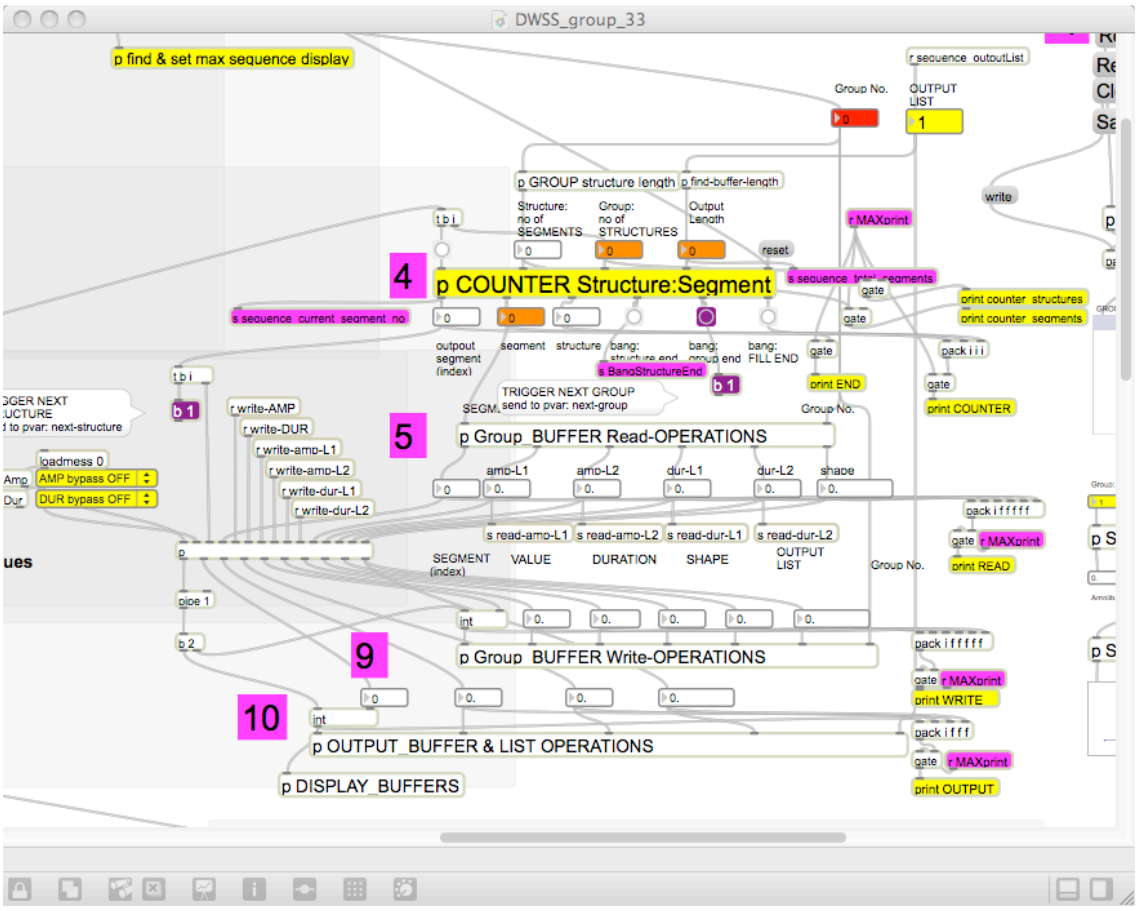

Figure 111: DWSS_group (detail j)

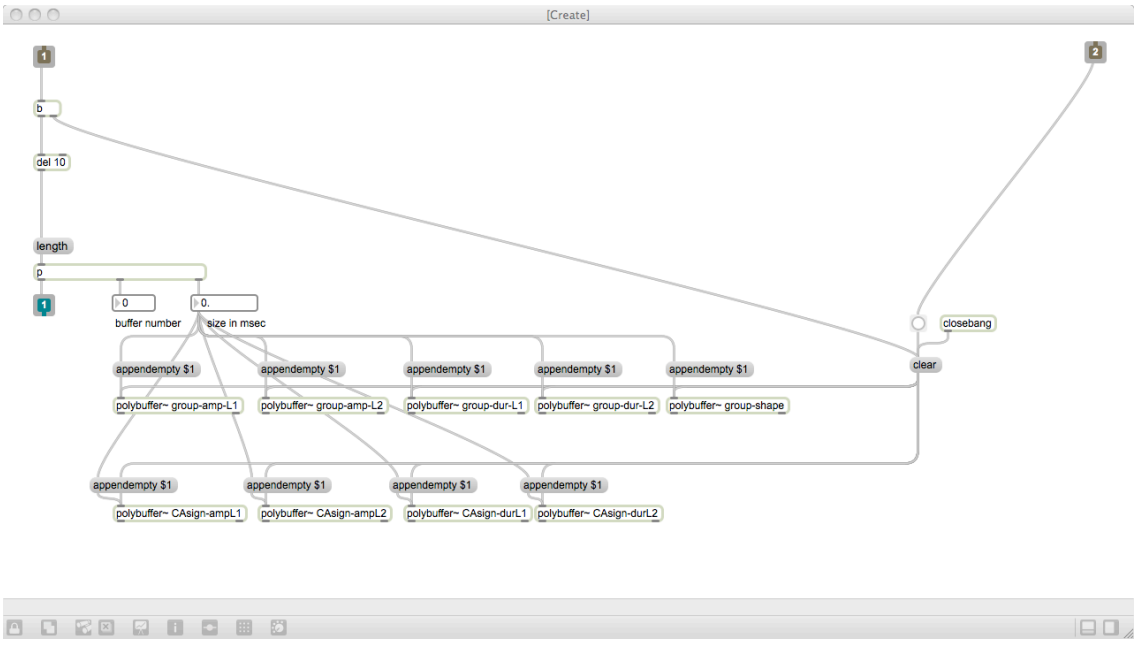

Figure 112: p Create Group Buffers

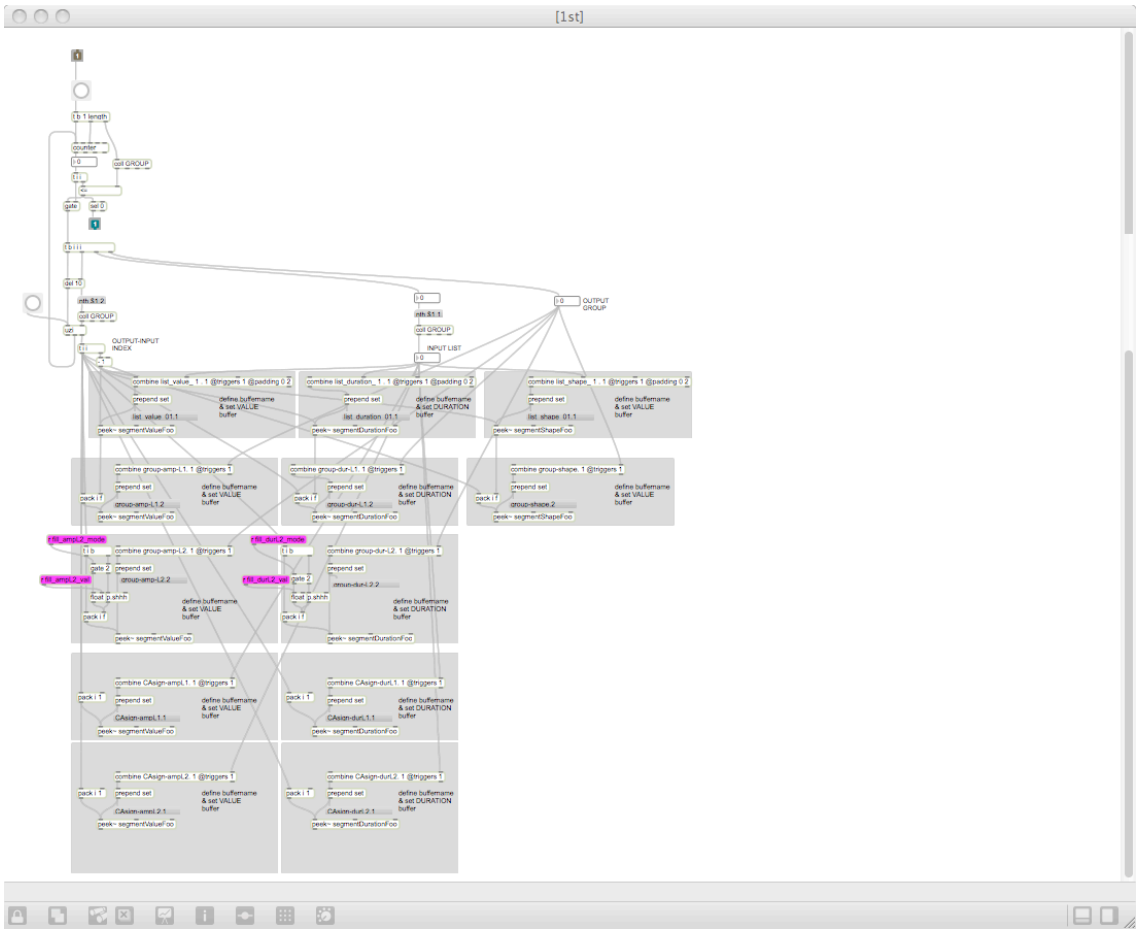

Figure 113: p 1st FILL Buffers

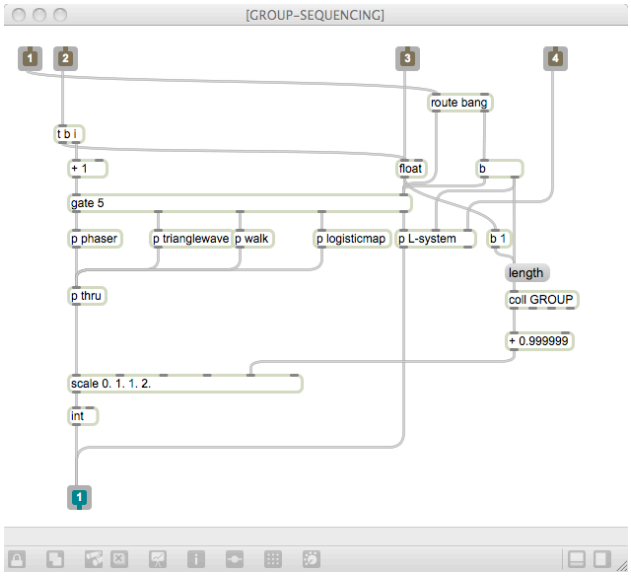

Figure 114: p GROUP-SEQUENCING

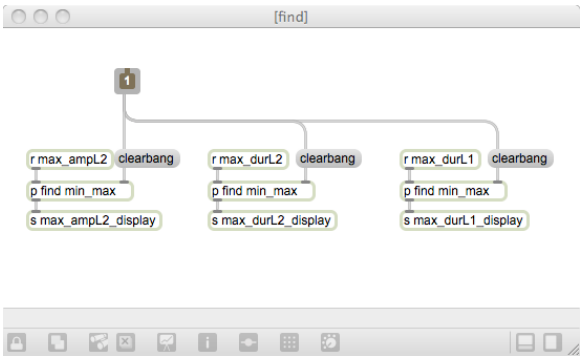

Figure 115: p find & set max sequence display

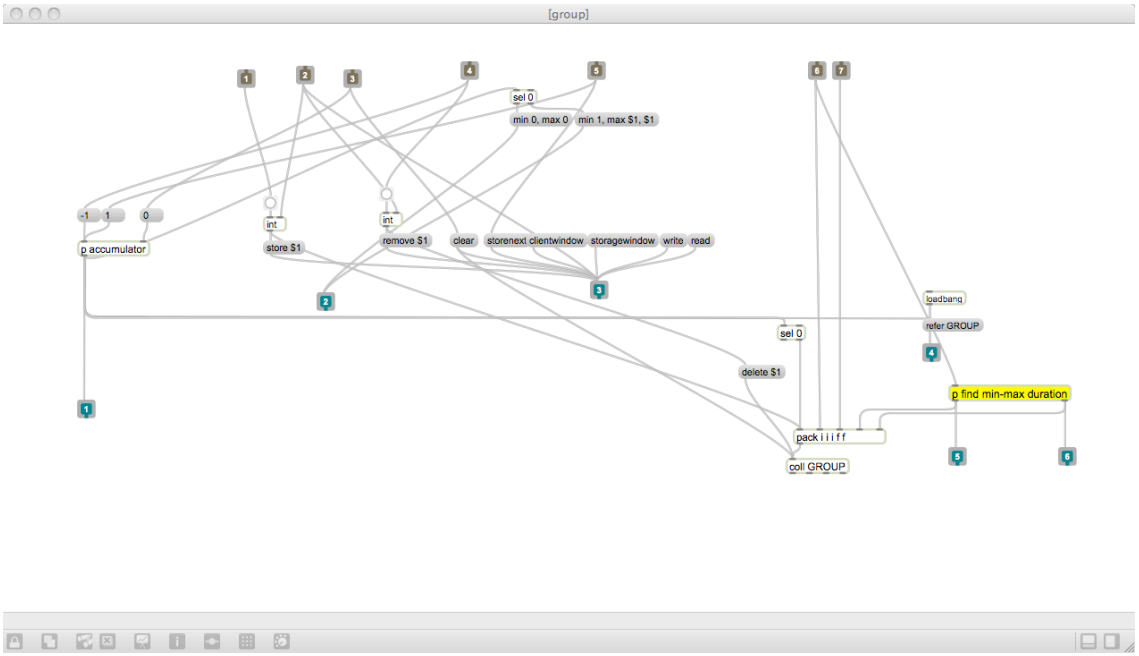

Figure 116: p group & preset operations

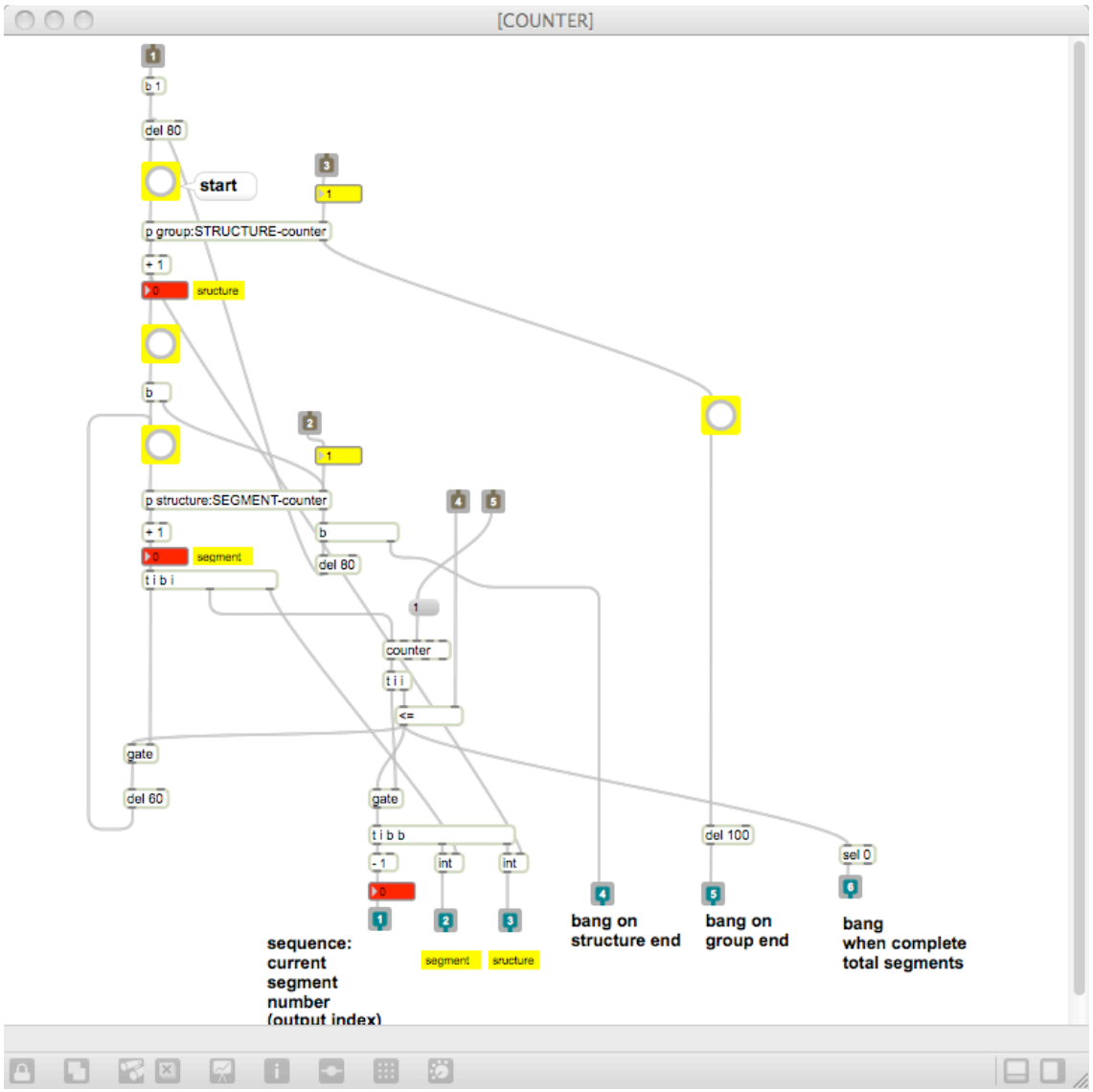

Figure 117: p COUNTER Structure:Segment

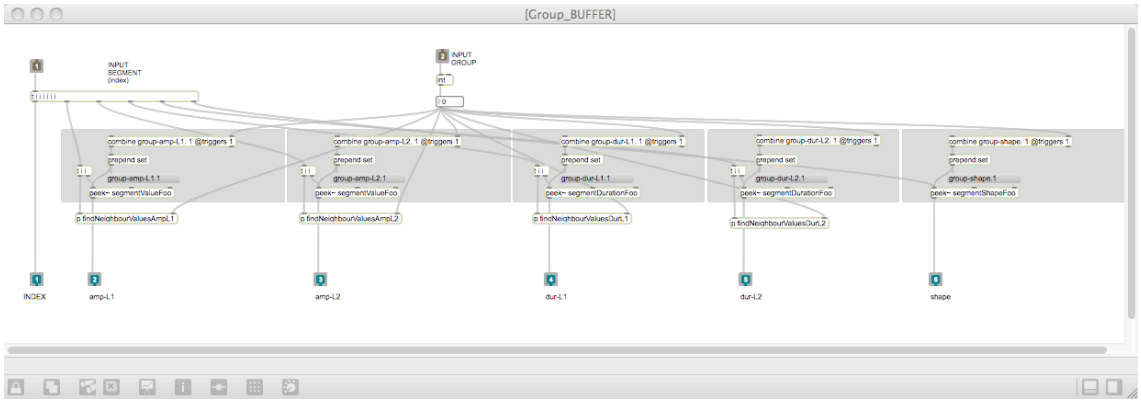

Figure 118: p Group_BUFFER Read-OPERATIONS

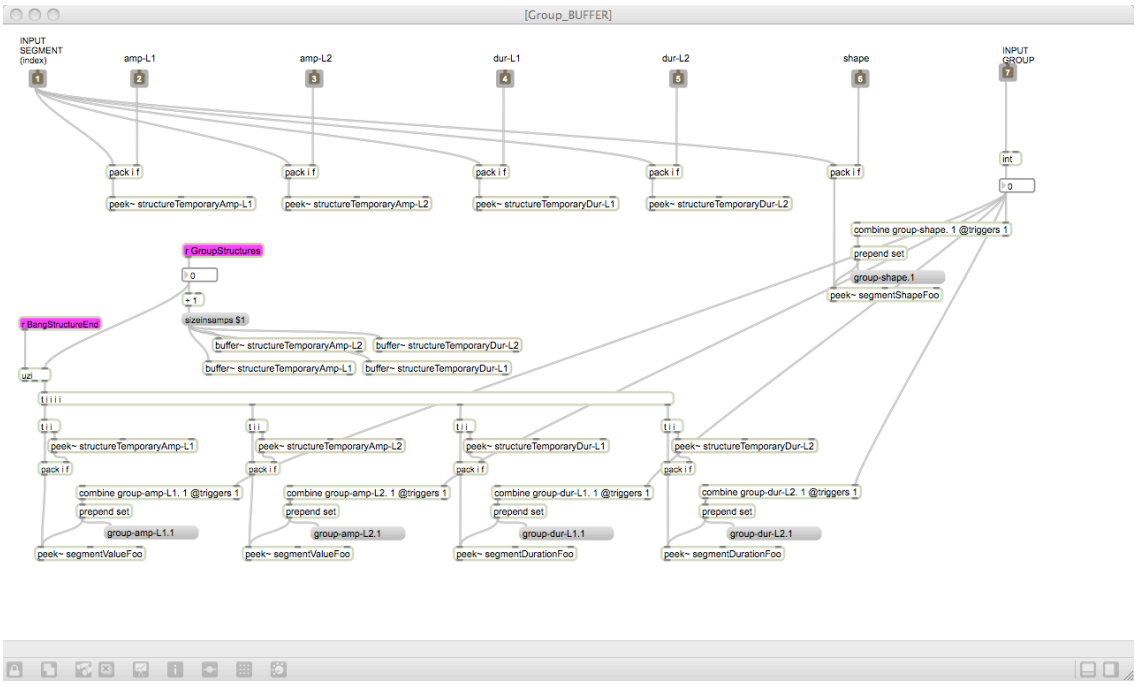

Figure 119: p Group_BUFFER Write-OPERATIONS

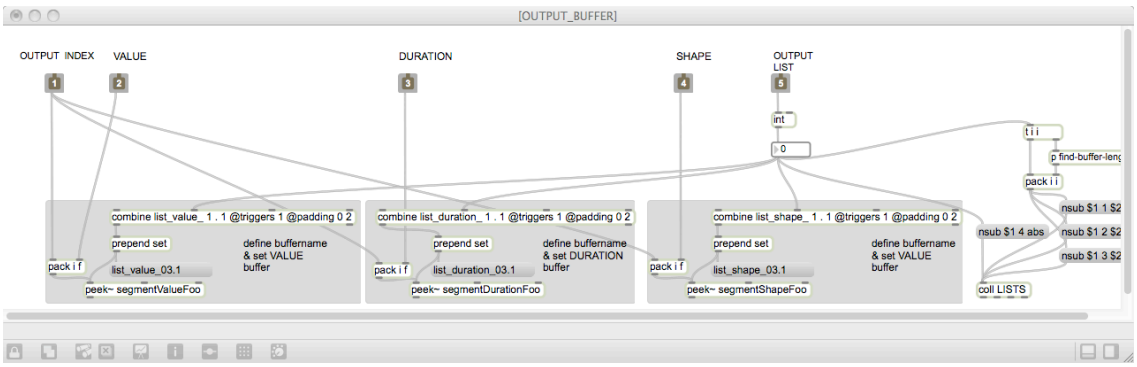

Figure 120: p OUTPUT_BUFFER & LIST OPERATIONS

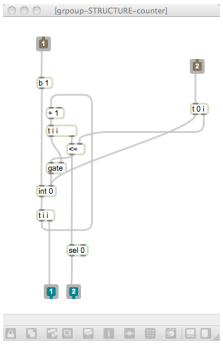

Figure 121: p grpoup-STRUCTURE-counter

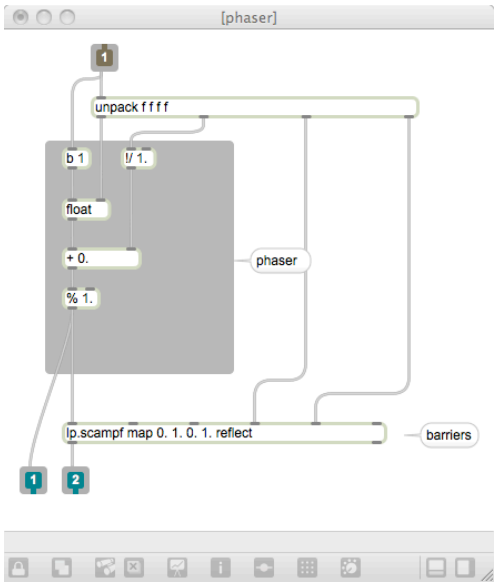

Figure 122: p phaser

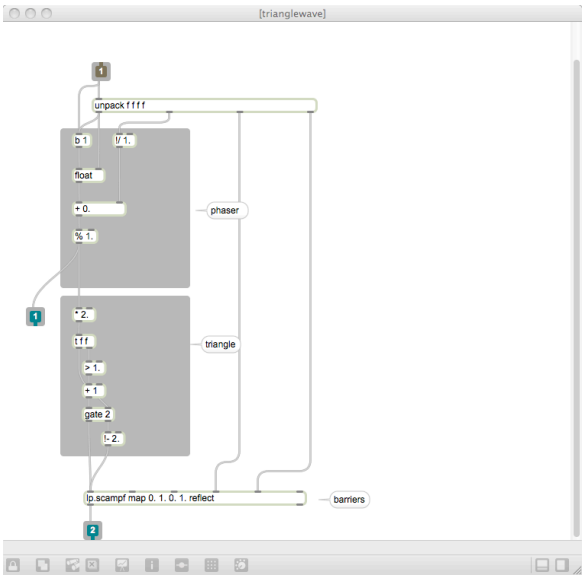

Figure 123: p trianglewave

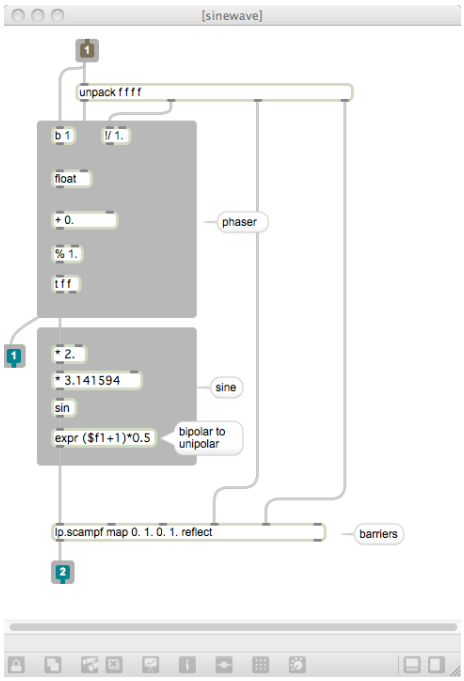

Figure 124: p sinewave

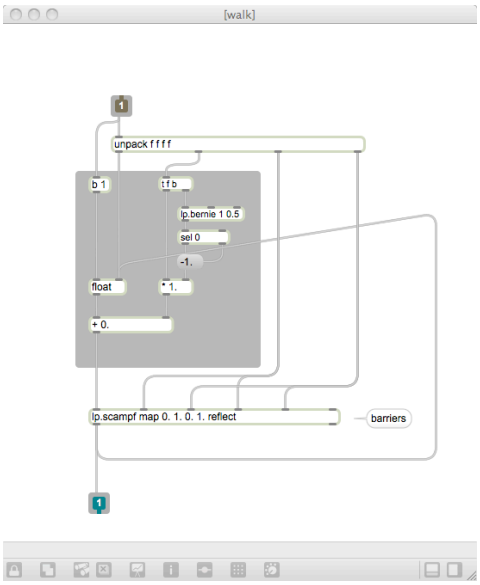

Figure 125: p walk

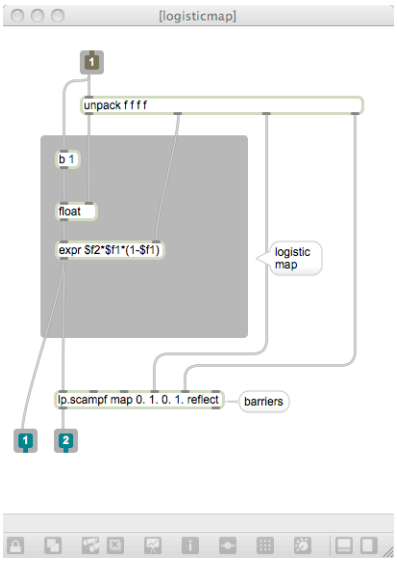

Figure 126: p logisticmap

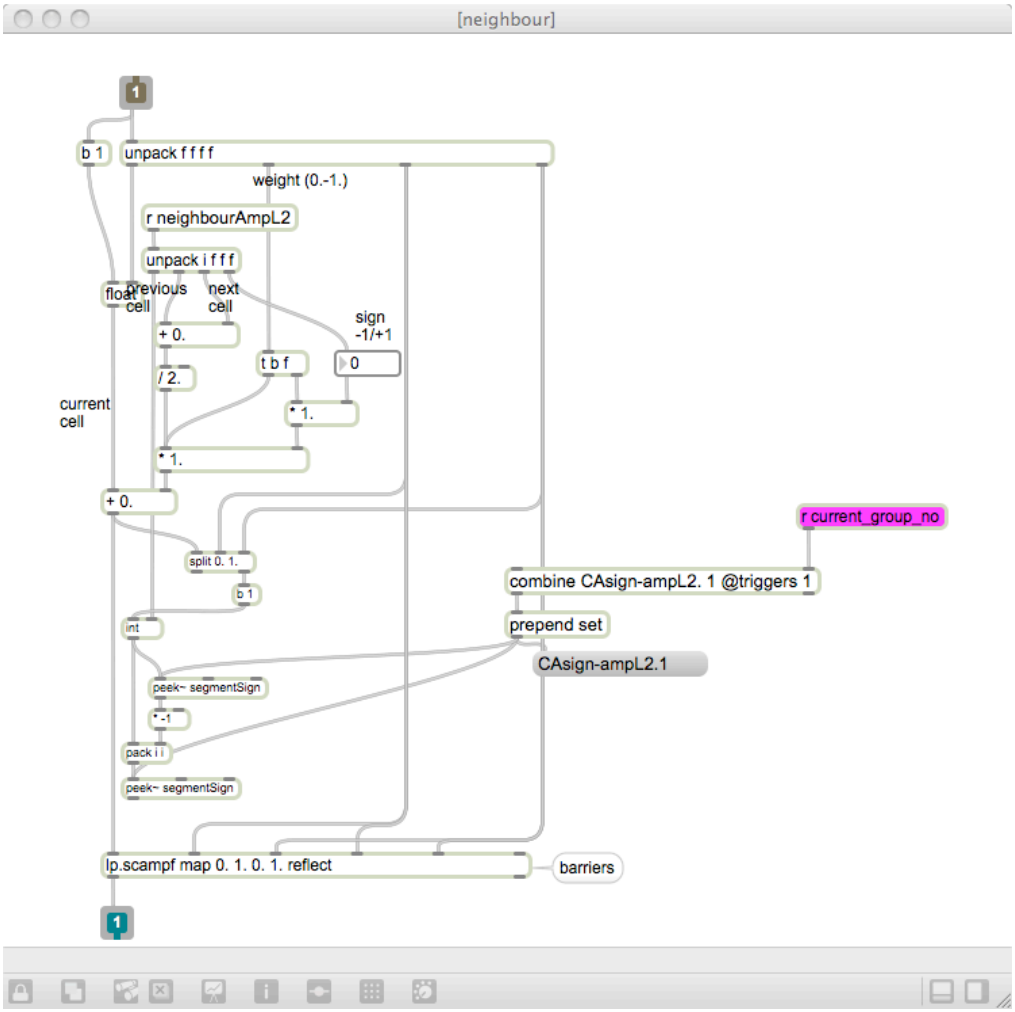

Figure 127: p neighbour

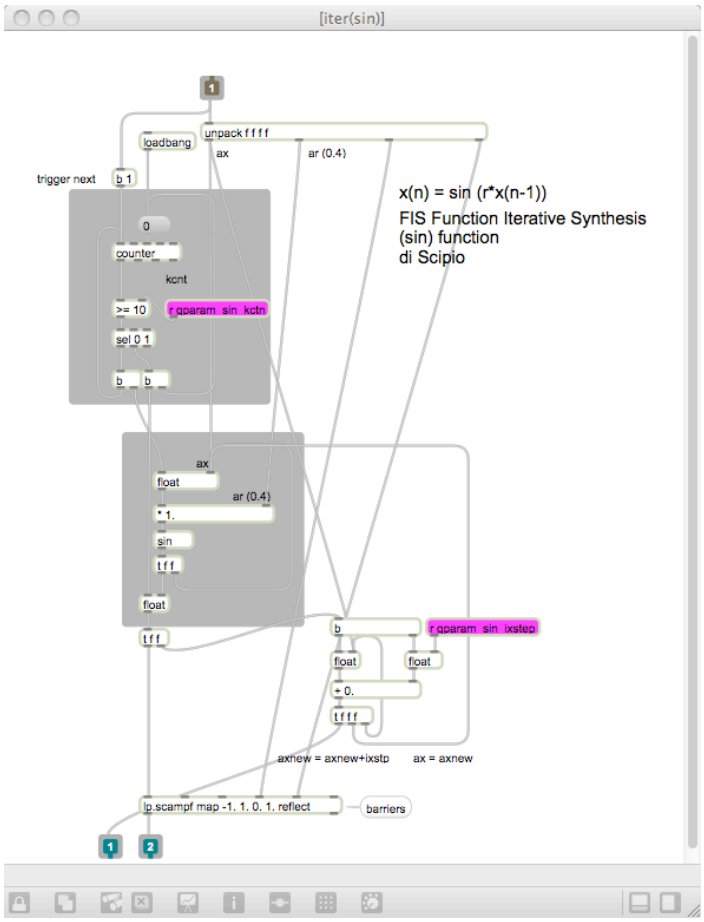

Figure 128: p iter(sin)

5) DWSS_sequence

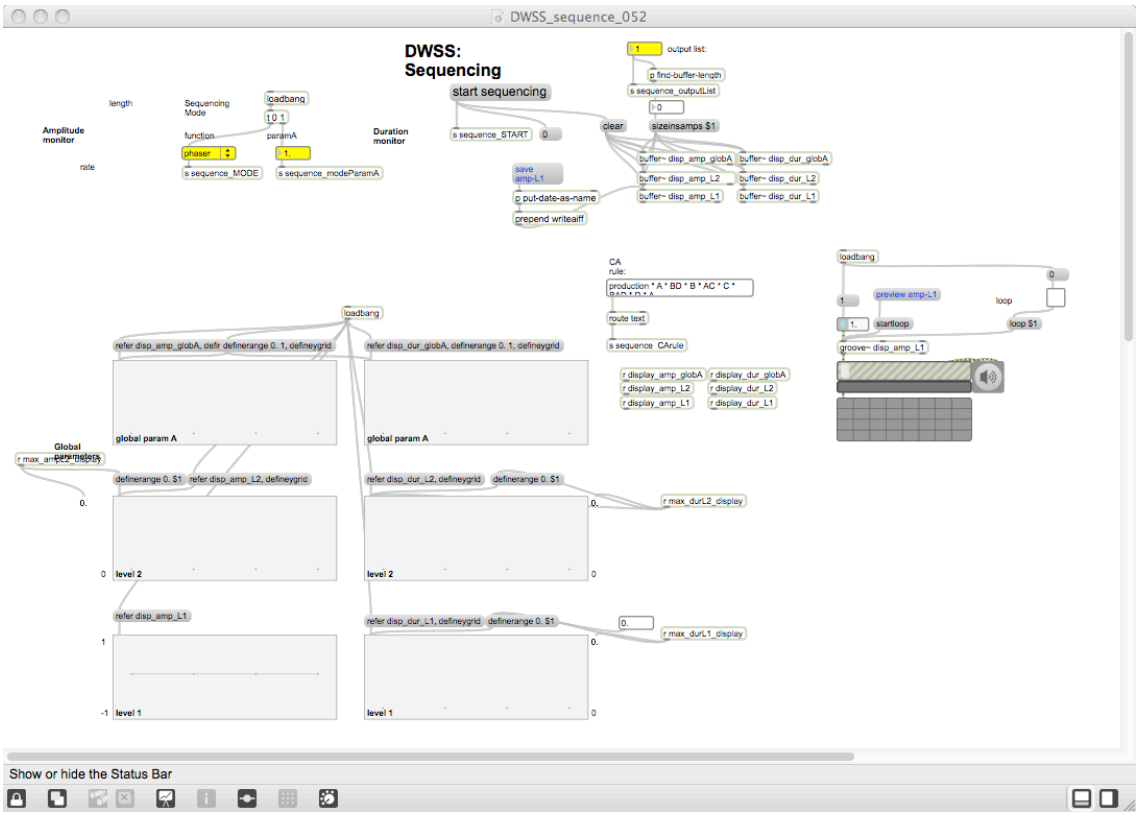

Figure 129: DWSS_sequence

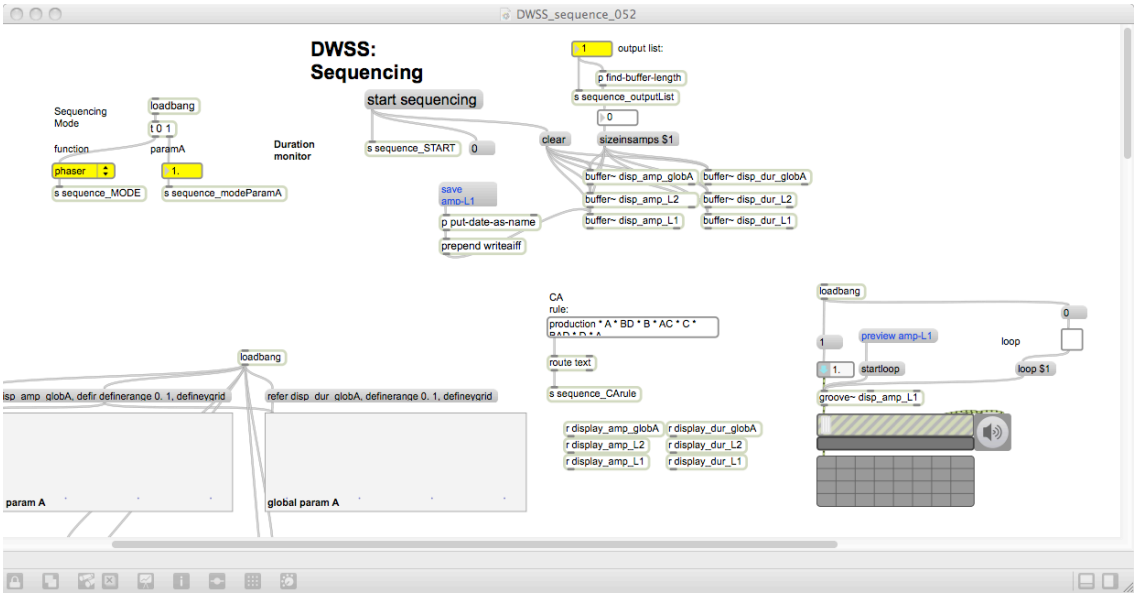

Figure 130: DWSS_sequence (detail a)

6) DWSS_synthesis

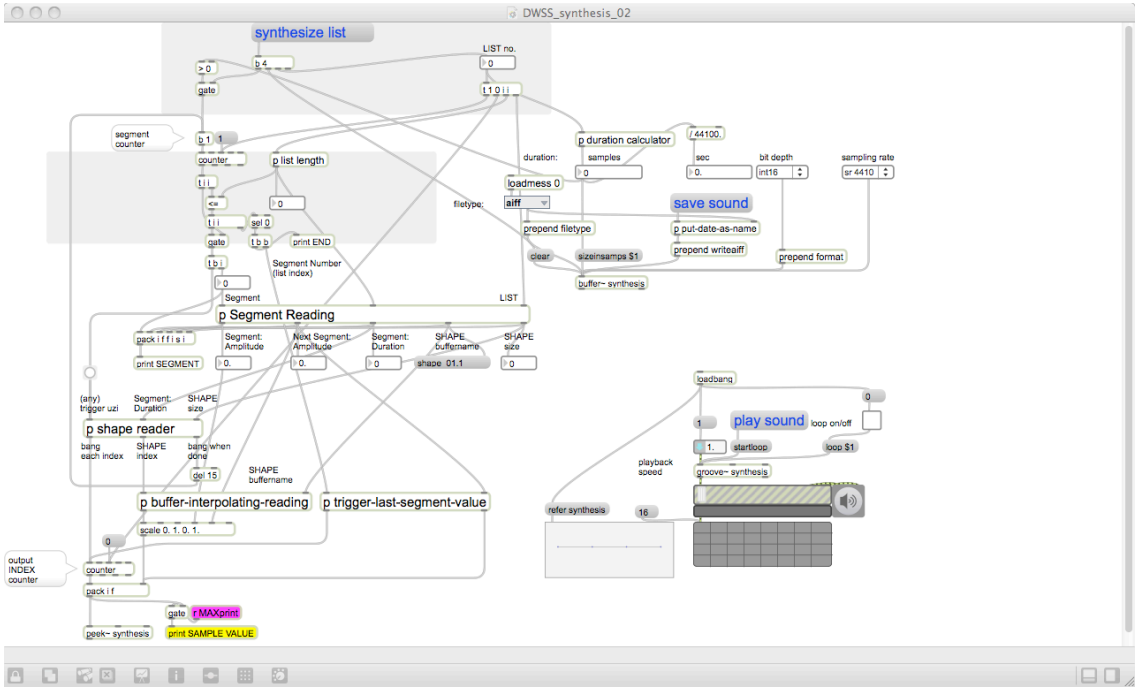

Figure 131: DWSS_synthesis

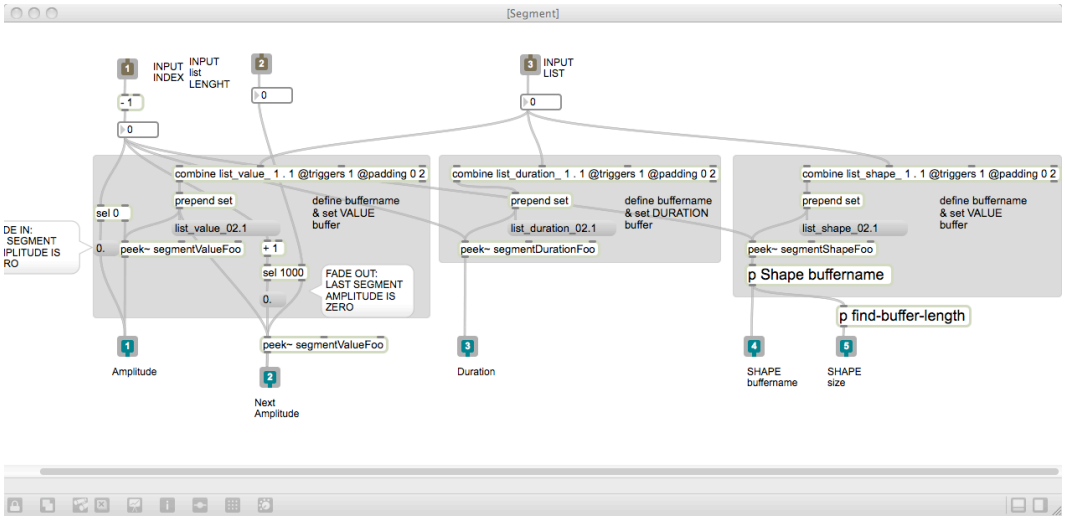

Figure 132: p Segment Reading

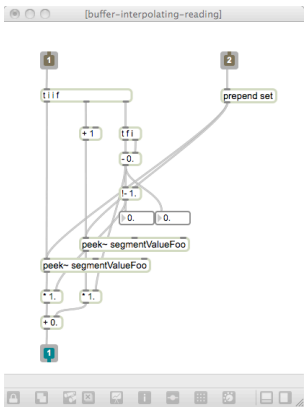

Figure 133: p buffer-interpolating-reading

Bibliography

Addison, P. 1996. Fractals and chaos: an illustrated course. CRC Press.

Adorno, T. 1997. Aesthetic theory. University of Minnesota Press.

Alexjander, S. 1994. Sequencia. Audio compact disc. Berkeley, California: Science and the Arts STA080.

———. 1999. "The infrared frequencies of DNA bases: science and art". IEEE Engineering In Medicine and Biology 18(2). http://www.healingmusic.org/SusanA/

Ames, C. 1983. “Stylistic Automata in Gradient.” In The Music Machine. C. Roads, ed. Cambridge: MIT Press.

———. 1987. “Automated Composition in Retrospect: 1956-1986.” Leonardo 20(2): 169- 185.

———. 1989. “The Markov Process as a Compositional Model: A Survey and Tutorial”, pp. 175-187, in: Leonardo 22:2, 1989.

———. 1990. “Statistics and Compositional Balance.” Perspectives of New Music 28(1): 80-111.

Ames, C. and M. Domino. 1992. “Cybernetic Composer: An Overview.” In Understanding Music with AI: Perspectives on Music Cognition. M. Balaban, K. Ebcioglu and O. Laske, eds. Cambridge: AAAI Press.

Andrews, I. 2002. "Post-digital aesthetics and the return to modernism". http://radioscopia.org/postdig.html

Ariza, C. 2005. "Navigating the Landscape of Computer-Aided Algorithmic Composition Systems: A Definition, Seven Descriptors, and a Lexicon of Systems and Research." In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 765-772.

———. 2007. "Automata Bending: Application of Dynamic Mutation and Dynamic Rules in Modular One-Dimensional Cellular Automata". Computer Music Journal, 31(1):29– 49.

———. 2008. "Python at the Control Rate: athenaCL Generators as Csound Signals". Csound Journal 9.

———. 2009. "Sonifying Sieves: Synthesis and Signal Processing Applications of the Xenakis Sieve with Python and Csound". ICMC Montreal 2009, ICMA

Assayag, G. and C. Rueda, M. Laurson, C. Agon, O. Delerue. 1999. "Computer-Assisted Composition at IRCAM: From PatchWork to OpenMusic". Computer Music Journal 23(3): 59-72.

Axen, U., and I. Choi. 1996. "Investigating Geometric Data with Sound". Proceedings of the 1996 International Conference on Auditory Display.

Badiou, A. 2004. Fifteen theses on contemporary art. Lacanian Ink, 23.

Bain, R. 1990. "Algorithmic Composition: Quantum Mechanics & the Musical Domain". Proceedings of the 1990 International Computer Music Conference. San Francisco: International Computer Music Association, pp. 276–279.

Balestrini, N. 1968. “Tape Mark I." in Cybernetic Serendipity. J. Reichardt, ed. London: Studio International, W. & J. Mackay. 55-56.

Banks, J. D, P. Berg, R. Rowe, and D. Theriault. 1979. “SSP — A Bi-Parametric Approach to Sound Synthesis.” In Sonological Reports. Utrecht: Institute of Sonology. 5.

Barlow, C. 1987. “Two Essays on Theory.” Computer Music Journal 11(1): 44-60.

Barnsley, M. 1988. Fractals Everywhere. Academic Press.

Bartetzki, A. 1997. “CMask, a Stochastic Event Generator for Csound.” Internet: http://gigant.kgw.tu-berlin.de/~abart/CMaskMan/CMask-Manual.htm.

Barthel-Calvet, A.S. 2001. “Chronologie.” Portrait(s) de Iannis Xenakis. Paris: Bibliothèque national de France. 25-80.

Bel, B. 1992. “Symbolic and Sonic Representations of Sound Object Structures.” In Understanding Music with AI: Perspectives on Music Cognition. M. Balaban, K. Ebcioglu and O. Laske, eds. Cambridge: AAAI Press / MIT Press. 65-109.

———. 1998. “Migrating Musical Concepts: An Overview of the Bol Processor.” Computer Music Journal 22(2): 56-64.

———. 2006. "The Bol Processor project: musicological and technical issues". Seminar of the Music, Informatics and Cognition research group, University of Edinburgh. (2006 October 31: Edinburgh, UK).

Bello, Angelo. 2000. “Simultaneous Spatialization and Synthesis of Sound with Nonlinear Functions”. Proc. Journées d’Informatique Musicales 2000. Bordeaux.

Bentley, P. J. and Corne, D. W. 2001. Creative Evolutionary systems. San Francisco, CA: Morgan Kaufmann

Berg, P. and R. Rowe, D. Theriault. 1980. “SSP and Sound Description.” Computer Music Journal 4(1): 25-35.

Berg, P. 1978. A User’s Manual for SSP. Utrecht: Institute of Sonology.

———. 1979a. “PILE - A Language for Sound Synthesis.” Computer Music Journal 3(1): 30-41.

———. 1979b. "SSP. A Bi-parametric Approach to Sound Synthesis". Sonological Reports. Institute of Sonology, Utrecht. 9-32.

———. 1996. “Abstracting the Future: The Search for Musical Constructs.” Computer Music Journal 20(3): 24-27.

———. 2003. Using the AC Toolbox. Den Haag: Institute of Sonology, Royal Conservatory.

———. 2009. Composing sound structures with rules. Contemporary Music Review, 28 (1).

Bernstein, A., and Cooper, E. 1976, "The Piecewise Linear technique of Electronic Music Synthesis", Journal of the audio Engineering Society, vol 24, 446-454.

Beyls, P. 1989. “The Musical Universe of Cellular Automata”, Proceedings of the International Computer Music Conference (ICMC), Columbus, OH, USA, 1989.

———. 2000. “Selectionist Musical Automata: integrating explicit instruction and evolutionary algorithms.” Proceedings of the International Computer Music Berlin 2000

———. 2004. "Cellular Automata Mapping Procedures", Proceedings of the International Computer Music Conference, Miami, USA, 2001

Bidlack, R. A. 1992. “Chaotic Systems as Simple (but Complex) Compositional Algorithms.” Computer Music Journal 16(3): 33-47.

Biles, J. A. 1994. “GenJam: A Genetic Algorithm for Generating Jazz Solos.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 131-137.

———. 2003. “GenJam in Perspective: A Tentative Taxonomy for GA Music and Art Systems.” Leonardo 36(1): 43-45.

Bilotta, E., S.Gervasi and P.Pantano. 2005. "Reading Complexity in Chua's Oscillator Through Music. Part I: A New Way of Understanding Chaos". International Journal of Bifurcation and Chaos, Vol. 15, No. 2. (2005), pp. 253-382

Bimber, B. 1990. “Karl Marx and the Three Faces of Technological Determinism.” Social Studies of Science 20(2): 333-351.

Birkhoff, G. D. 1933. Aesthetic Measure. Cambridge: Harvard University Press.

Blum, T. 1979. “Herbert Brü n: Project Sawdust.” Computer Music Journal 3(1): 6-7.

Boersma, Paul & David Weenink 2009. Praat: doing phonetics by computer (Version 5.1.05) [Computer program]. Retrieved May 1, 2009, from http://www.praat.org/

Boulanger, Richard Charles. 2000. The Csound Book: Perspectives in Software Synthesis, Sound Design, Signal Processing, and Programming. Cambridge: MIT Press.

Bogaards, N., A. Röbel, and X. Rodet. 2004. "Sound Analysis and Processing with Audiosculpt 2." Proceedings of the 2004 International Computer Music Conference. San Francisco, California: International Computer Music Association, pp. 462-465.

Bracewell, R.N., The Fourier Transform and Its Applications (McGraw-Hill, 1965, 2nd ed. 1978, revised 1986)

Brinkman, A. 1981. “Data Structures for a Music-11 Preprocessor.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association.

Brown, A. 2005. Extending dynamic stochastic synthesis. International Computer Music Conference, Barcelona.

Brü n, H. 1968. “Composition with Computers.” In Cybernetic Serendipity. J. Reichardt, ed. London: Studio International, W. & J. Mackay. 20.

———. 1969. “Infraudibles.” In Music by Computer. H. von Foerster and J. W. Beauchamp, eds. New York: John Wiley & Sons. 117-120.

———. 1970. “From Musical Ideas to Computers and Back.” In The Computer and Music. H. B. Lincoln, ed. Ithaca: Cornell University Press. 23-36.

———. 1970. When Music Resists Meaning, chapter From Musical Ideas to Computers and Back. Wesleyan University Press, 2004.

———. 1971. “Technology and the Composer.” In Music and Technology (Proceedings of the Stockholm Meeting organized by UNESCO). Paris: La Revue Musicale. 181-192.

———. 1980. “Dust, More Dust, Dustiny.” In UNESCO Computer Music: Report on an internationl project including the international workshop held at Aarhus, Denmark in 1978. M. Battier and B. Truax, eds. Canadian Commission for UNESCO. 85-86.

Brü n, H. and A. Chandra. 2001. "A Manual for SAWDUST.". http://grace.evergreen.edu/~arunc/brun/sawdust

Burks, A. (Ed) 1970. "Essays on Cellular Automata". Univ. of Illinois Press.

Burraston, D., E. Edmonds, D. Livingstone, and E. Miranda. 2004. "Cellular Automata in MIDI based Computer Music". Proceedings of the 2004 International Computer Music Conference.

Burraston, D. and Edmonds, E. 2005. "Cellular Automata in Generative Electronic Music and Sonic Art: A Historical and Technical Review." Digital Creativity 16(3): 165-185, Taylor and Francis.

Burns, K. H. 1994. The History and Development of Algorithms in Music Composition, 1957-1993. D.A. Dissertation, Ball State University.

Burt, W. 1996. “Some Parentheses Around Algorithmic Composition.” Organised Sound 1(3): 167-172.

Buxton, W. 1975. Manual for the POD Programs. Utrecht: Institute of Sonology, University of Utrecht.

———. 1978. Design Issues in the Foundation of a Computer-Based Tool for Music Composition. Toronto: Technical Report Computer Systems Research Group.

Buxton, W., Patel, S., Reeves, W., and Baecker, R Object and the Design of Timbral Resources. Computer Music Journal, 6(2) (1982), 32-44.

Buxton, W. and W. Reeves, R. Baecker, L. Mezei. 1978. "The Use of Hierarchy and Instance in a Data Structure for Computer Music." Computer Music Journal 2(4): 10-20.

Cascone, K. 2000. The aesthetics of failure: post-digital tendencies in contemporary computer music. Computer Music Journal 24(4): 12–18.

Castagné, N. et Cadoz, C. "GENESIS: A Friendly Musician-Oriented Environment for Mass- Interaction Physical Modeling", Proceedings of the International Computer Music Conference, San Francisco, International Computer Music Association, 2002.

Chadabe, J. 1997. Electric Sound: The Past and Promise of Electronic Music. New Jersey: Prentice-Hall.

Chandra, A. 1993 "CounterWave: a program for controlling degrees of independence between simultaneously transforming waveforms" in Proceedings of the Inter national Association of Knowledge Technology and the Arts, Osaka, Japan: September.

———. 1994. The linear change of waveform segments causing non-linear changes of timbral presence. Contemporary Music Review 10(2): 157–69.

Chang, J. 1999. Composing Noise. Institute of Sonology.

Chapman, D., Clarke, M., Smith, M., Archbold, P. 1996. "Self-similar Grain Distribution: a Fractal Approach to Granular Synthesis". In Proceedings of ICMC, Hong Kong, pp212-213, ICMA, San Francisco, 1996.

Chareyron, J. 1990. "Digital Synthesis of Self-Modifying Waveforms by Means of Linear Automata." Computer Music Journal 14(4):25-41

Chion, M. 1982. La musique électroacoustique. Paris: Presses universitaires de France.

———. 1983. Guide des Objets Sonores. Eds. Buchet/Chastel, Paris. 1995 translation by John Dack/Christine North. Accessed (13/8/2010): http://www.ears.dmu.ac.uk/spip.php?rubrique219

Chion, M. 1994. "The Three Listening Modes," in Audio/Vision: Sound on Screen (New York: Columbia University Press, 1994), 25-34.

Chomsky, N. 1957. Syntactic Structures. The Hague: Mouton.

Chowning, J. 1973. "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation". Journal of the Audio Engineering Society 21 (7

Chowning, J. and Bristow, D. 1986. FM Theory and Applications for Musicians. Tokyo: Yamaha Music Foundation.

Ciamaga, G. 1975. “The Tape Studio.” In The Development and Practice ofElectronic Music. J. H. Appleton and R. C. Perera, eds. Englewood Cliffs: Prentice-Hall. 68-137.

Clarke, M. 1996. Composing at the intersection of time and frequency. Organised Sound 1(2): 107–17.

Clifton, Thomas. 1983. Music as Heard: A Study in Applied Phenomenology. New Haven and London: Yale University Press.

Gogins, M. 1991. “Iterated functions systems music,” Computer Music Journal, vol. 15, no. 1, pp. 40–48.

———. 2009. "Score Generating Lindenmayer Systems In The Generalized Contextual Group". Internet: http://michael-gogins.com/pdf/Lindenmayer_Systems_Based_on_Riemannian_Transformations.pdf

Cohen, J. E. 1962. “Information Theory and Music.” Behavioral Science 7(2): 137-163.

Collins, N. 1999. "SplineSynth: An Interface to Low-Level Digital Audio". Proceedings of the Diderot Forum on Mathematics and Music, Vienna, ISBN 3-85403-133-5, pp 49-61.

———. 2000. "SplineSynth2: Interpolating Break-Point Sets To Obtain Sound Transformations Distinct From a Cross-Fade". Unpublished research report on the SplineSynth2 software.

———. 2008. Errant Sound Synthesis. International Computer Music Conference, Belfast.

———. 2008. The analysis of generative music programs. Organised Sound, 13, 237–248.

———. 2009. 'Musical Form and Algorithmic Composition', Contemporary Music Review, 28: 1, 103 — 114

Collins, N. and d'Escrivan, J. (eds.). 2007. The Cambridge Companion to Electronic Music. Cambridge: Cambridge University Press

Cope, D. 1991. Computers and Musical Style. Oxford: Oxford University Press.

———. 1993. "Algorithmic Composition [re]Defined." In Proceedings ofthe International Computer Music Conference. San Francisco: International Computer Music Association. 23- 25.

———. 1996. Experiments in Music Intelligence. Madison, WI: A-R Editions.

———. 2000. The Algorithmic Composer. Madison, WI: A-R Editions.

Cranfield, B. 2002. Producing noise: Oval and the politics of digital audio. Parachute no. 107: 42–51.

Dahlhaus, C. 1982. Esthetics of music. Cambridge & New York: Cambridge University Press.

Dannenberg, R. B. 1997. "The Implementation of Nyquist, A Sound Synthesis Language." Computer Music Journal 21(3): 71-82.

Degazio, B. 1997. "The Evolution of Musical Organisms." Leonardo Music Journal 7: 27-33.

De Poli, G., A. Piccialli, and C. Roads, eds. 1991. "Representations of Musical Signals." MIT Press.

Delatour, T. 2000. Molecular music: the acoustic conversion of molecular vibrational spectra. Computer Music Journal 24(3): 48–68.

Deguet, J., Y. Demazeau and L. Magnin, 2005. “Elements about the emergence issue: A survey of emergence definitions.” Proceedings of ECCS’05, European Conference on Complex Systems Paris, 14-18 November 2005.

Dewey, J. 1934. Art as Experience. Reprint edition by Perigee Books, The Berkley Publishing, Group, New York, 1980

Di Scipio, A. 1994a. Micro-time sonic design and timbre formation. Contemporary Music Review 10(2): 135–48.

———. 1994b. “Formal Processes of Algorithmic Composition Challenging the Dualistic Paradigm of Computer Music.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 202-208.

———. 1995a. “On Different Approaches to Computer Music as Different Models of Compositional Design.” Perspectives of New Music 331-2: 360-402.

———. 1995b. Inseparable models of materials and of musical design in electroacoustic and computer music. Journal of New Music Research 24(1): 34– 50.

———. 1996. Functional iteration synthesis: A revitalization of non-standard synthesis. Journal of new music research 25(1).

———. 1997. “Towards a Critical Theory of (Music) Technology. Computer Music and Subversive Rationalization.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 62-65.

———. 1998. “Compositional Models in Xenakis’s Electroacoustic Music.” Perspectives of New Music 36, no. 2: 201-243

———. 1998. Questions concerning music technology. Angelaki, 3 (2).

———. 2000. Sound synthesis by iterated non-linear functions. In Virtual Sound: Sound Synthesis and Signal Processing, Theory and Practice with Csound, pp. 385–97. Rome, Italy: ConTempo.

———. 2002a. “The Synthesis of Environmental Sound Textures by Iterated Nonlinear Functions, and its Ecological Relevance to Perceptual Modeling”, Journal of New Music Research, 31(2):109-117.

———. 2002b "Systems of embers, dust, and clouds: Observations after xenakis and brü n". Computer Music Journal, 26(21).

Di Scipio, A. and Prignano, I. 1996 Synthesis by functional iterations. a revitalization of nonstandard synthesis. Journal of New Music Research 25(1) 31–46.

Dobereiner, L. 2009a. "PV Stoch: A Spectral Stochastic Synthesis Generator", In Proceedings of the Sound and Music Computing Conference '09 (SMC '09), Porto.

———. 2009b. Compositionally Motivated Sound Synthesis, In Proceedings of next generation 3.0, ZKM Karlsruhe, 2009

———. 2011. Models of Constructed Sound: Nonstandard Synthesis as an Aesthetic Perspective, Computer Music Journal 35(3): 28–39

Dobson, R. and Fitch, J. 1995. “Experiments with Chaotic Oscillators”, Proceedings of the 1995 International Computer Music Conference, 45-48. San Francisco, CA: International Computer Music Association (ICMA).

Dodge, C. 1988. “Profile: A musical fractal.” Computer Music Journal 12(3): 10-14.

Dodge, C. & Jerse, T. A. (1997) Computer music: synthesis, composition and performance. N.Y.: Schirmer.

Doornbusch, P. 2004. “Computer Sound Synthesis in 1951: The Music of CSIRAC.” Computer Music Journal 28(1): 10-25.

Eco, U. 1989. The Open Work. Translated by A. Cancogni. Cambridge: Harvard University Press.

Edwards, M. 2009. "Algorithmic Composition: Computational Thinking in Music", School of Arts, Culture and Environment University of Edinburgh Edinburgh, UK, accessed: http://people.ace.ed.ac.uk/staff/medward2/algorithmic-composition.pdf (22/5/2011)

Englert, G. 1981. “Automated Composition and Composed Automation.” Computer Music Journal 5(4): 30-35.

Essl, G. 2006. "Circle maps as a simple oscillators for complex behavior: II. Experiments" In Proceedings of the International Conference on Digital Audio Effects (DAFx), Montreal, September 18-20, 2006.

Essl, K. 2007. “Algorithmic Music.” In The Cambridge Companion to Electronic Music, Cambridge: Cambridge University Press, edited by Nick Collins and Julio d’Escriván

Evans, B. 2000. “Hearing the Mandelbrot Set,” The Csound Book, ed. R. Boulanger, (2000), MIT Press: Cambridge.

Fact Index: Noise Music. 2004. http://www.fact-index.com/ n/no/noise_music.html, visited 9 September 2004.

Feenberg, A. (1990). The ambivalence of technology. Sociological Perspectives, 33(1).

———. (1991). A critical theory of technology. Oxford University Press.

Feller, W. 1968. An Introduction to Probability Theory and its Applications (Volume 1)

Flake, W.G., 1998. The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation, Cambridge, MA: MIT Press.

Freed, A. and X. Rodet, Ph. Depalle. 1993. Performance, Syn thesis and Control of Additive Synthesis on a Desktop Computer Using FFT. Proc. ICMC, 1993.

Fry, C. 1980. “Flavors Band: A Language for Specifying Musical Style.” In The Music Machine. C. Roads, ed. Cambridge: MIT Press. 295-309.

Garton, B 1997. RTcmix - Using CMIX in Real Time, In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association

Gabor, D. 1947. Acoustical quanta and the theory of hearing. Nature 159(4044): 591–4.

Ganguly, N Sikdar, B. Deutsch, A. Canright, G. Chaudhuri, P. 2003. “A survey on cellular automata”, Technical report, Centre for High Performance Computing, Dresden University of Technology.

Geiger, G. 2006. "Table lookup oscillators using generic iIntegrated wavetables", in Proc. Conf. on Digital Audio Effects (DAFX-06), Montreal, Canada, Sept. 2006

Gleick, J. 1987. Chaos: Making a New Science, Viking Penguin.

———. 2011, The Information: A History, a Theory, a Flood, New York: Pantheon

Gogins, M. 1991. “Iterated Functions Systems Music”, Computer Music Journal 15, March 1991.

———. 1995. “Gabor Synthesis of Recurrent Iterated Function Systems”, Proceedings of the International Computer Music Conference, September 1995. Goldberg, S. 1986. "Probability: An Introduction", New York: Dover.

Goldstein, J. 1999, "Emergence as a Construct: History and Issues", Emergence: Complexity and Organization 1 (1): 49–72

Gottfried M-K, I.Choi, N.Weber, R.Bargar. 1993." Musical signals from Chua’s circuit", IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 40

Grossman, G. 1987. "Instruments, Cybernetics and Computer Music", Proceedings of the 1987 International Computer Music Conference, San Francisco: ICMC, pp. 212-219.

Hedelin, F. 2008. Formalising form: An alternative approach to algorithmic composition. Organised Sound, 13(3), 249–257.

Hamman, M. 1994. “Dynamically Configurable Feedback/Delay Networks: A Virtual Instrument Composition Model,” Proceedings of the 1994 International Computer Music Conference. San Francisco: ICMA.

Hamman, M. 1995. "Computation as Mediation in Composition - From the Technical to the Technological". <http://www.shout.net/~mhamman> [accessed: 17/05/2011]

———. 1999a. "Structure as Performance: Cognitive Musicology and the Objectification of Compositional Procedure." In Otto Laske: Navigating New Musical Horizons. Edited by Jerry Tabor. 37-52.

———. 1999b. "From Symbol to Semiotic: Representation, Signification and the Composition of Music Interaction." Journal of New Music Research 28.2:90-104.

———. 2000. "Priming Computer-Assisted Music Composition through Design of Human/Computer Interaction," Mathematics and Computers in Modern Science, ed. N. E. Mastorakis. World Scientific Engineering Society

———. 2002. "From Technical to Technological: The Imperative of Technology in Experimental Music Composition." Perspectives in New Music 40.1:92-120.

Harley, J. 1994. “Algorithms Adapted From Chaos Theory.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 209-212.

———. 1995. "Generative Processes in Algorithmic Composition: Chaos and music." Leonardo Music Journal 28.3.

———. 2002 “The Electroacoustic Music of Iannis Xenakis.” Computer Music Journal 26(1): 33-57

———. 2004. Xenakis: his life in music. London: Taylor & Francis Books

Herman, Martin. 1993. "Deterministic Chaos, Iterative Models, Dynamical Systems and Their Application in Algorithmic Composition." Proceedings of the International Computer Music Conference 194-197.

Hiller, L. 1956. “Abstracts: Some Structural Principles of Computer Music.” Journal of the American Musicological Society 9(3): 247-248.

———. 1970. “Music Composed with Computers: An Historical Survey.” In The Computer and Music. H. B. Lincoln, ed. Ithaca: Cornell University Press. 42-96.

———. 1981. “Composing with Computers: A Progress Report.” Computer Music Journal 5(4): 7-21.

Hiller, L. and L. Isaacson. 1958. “Musical Composition with a High-Speed Digital Computer.” Journal of the Audio Engineering Society 6(3): 154-160.

———. 1959. Experimental Music. New York: McGraw-Hill.

Hinojosa C. R. 2003. "Realtime Algorithmic Music Systems From Fractals and Chaotic Functions: Toward an Active Musical Instrument ." PhD Thesis, Universitat Pompeu Fabra.

Hoffman, P. 1998. "Evaluating the dynamic stochastic synthesis." Publications du Laboratoire de Mecanique et d'Acoustique 148: F4.1-F4.8.

———. 2000. “A New GENDYN Program.” Computer Music Journal 24(2): 31-38.

———. 2001. “Analysis through Resynthesis.” Presences of Iannis Xenakis. Paris:CDMC: 185-194.

———. 2002. “Towards an ‘Automated Art’: Algorithmic Processes in Xenakis’ Compositions.” Contemporary Music Review 21(2-3): 121-131.

———. 2004. “‘Something rich and strange’: Exploring the Pitch Structure of GENDY3.” Journal of New Music Research 33(2): 137-144.

———. 2009. “Music Out of Nothing? A Rigorous Approach to Algorithmic Composition by Iannis Xenakis.” PhD diss., Technischen Universitä t Berlin.

———. 2011. “Xenakis Alive!” Explorations and extensions of Xenakis’ electroacoustic thought by selected artists". Proceedings of the Xenakis International Symposium, London.

Horner, A. 2003. "Auto-Programmable FM and Wavetable Synthesizers," Contemporary Music Review, 22(3), 21-29.

Holtzman, S.R. 1978. “A Description of an Automatic Digital Sound Synthesis Instrument.” D.A.I. Research Report No. 59. Edinburgh: Department of Artificial Intelligence

———. 1980. “A Generative Grammar Definition Language for Music.” Interface 9: 1-47.

———. "Using Generative Grammers for Music Composition." Computer Music Journal 5.1:51-64. {366-0.0}

———. 1994. Digital Mantras: the Languages of Abstract and Virtual Worlds. Cambridge, MA: MIT Press.

———. 1997. Digital Mosaics: the Aesthetics of Cyberspace. New York: Simon and Schuster.

Horner, A. 1998 “Nested modulator and feedback FM matching of instrument tones”. IEEE Trans- actions on Speech and Audio Processing 6(4), 398–409.

Jacob, B. 1996. “Algorithmic Composition as a Model of Creativity.” Organised Sound 1(3): 157-165.

Jaffe, David A. Ten criteria for evaluating synthesis techniques. Computer Music Journal, 19(1):76-87, 1995

Johnson, R. S. 2006. "Composing with Fractals". In J. Fauvel, R. Flood and R. Wilson, eds., Music and Mathematics, 2006.

Jones, K. 1981. “Compositional Applications of Stochastic Processes.” Computer Music Journal 5(2): 45-61.

———. 1995. “The Algorithmic Muse: New Listening Paradigms and the Harmonies of Chaos.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 19-22.

Kane, B. 2007, L’Objet Sonore Maintenant: Pierre Schaeffer, sound objects and the phenomenological reduction, Organised Sound 12(1): 15-24, Cambridge University Press.

Karpus, K. and A. Strong. 1983. “Digital Synthesis of Plucked-String and Drum Timbres.” Computer Music Journal 7(2): 43-55.

Keller, D., and Truax, B. 1998. Ecologically-based granular synthesis. Proc. of the 1998 Int. Computer Music Conf., pp. 117–20. San Francisco: Computer Music Association.

Kirke, A. and Miranda, E. R. 2007. “Capturing the aesthetic: Radial mappings for cellular automata music” Journal of the ITC Sangeet Research Academy, 21, pp. 15-23.

Koenig, G. M. 1959. Studium im Studio. Die reihe #5, English edition 1961, pp. 30–9. Bryn Mawr: Theodore Presser Company.

———. 1969. “Project One.” In Electronic Music Report. Utrecht: Institute of Sonology. 2: 32-46.

———. 1970. “Project Two - A Programme for Musical Composition.” In Electronic Music Report. Utrecht: Institute of Sonology. 3.

———. 1970b. “The Use of Computer Programs in Creating Music.” In Music and Technology (Proceedings of the Stockholm Meeting organized by UNESCO). Paris: La Revue Musicale.

———. 1971a. Summary Observations on Compositional Theory. Utrecht: Institute of Sonology.

———. 1979. PROTOCOL: A Report of the 1974/75 Class in Programmed Music at the Institute of Sonology. Utrecht: Institute of Sonology, University of Utrecht.

———. 1980a. “Composition Processes.” In UNESCO Computer Music: Report on an international project including the international workshop held at Aarhus, Denmark in 1978. M. Battier and B. Truax, eds. Canadian Commission for UNESCO. 105-126.

———. 1980b. PRIXM Manual. Utrecht: Institute of Sonology, University of Utrecht.

———. 1987. Genesis of form in technically conditioned environments. Interface 16(3): 165–76.

———. 1983. “Aesthetic Integration of Computer-Composed Scores.” Computer Music Journal 7(4): 27-32.

———. 1991. "Working with 'Project One': My Experiences with Computer Composition." Interface 20.3-4:175-180.

———. 1992. "Segmente: A structural landscape." Interface [Journal of New Music Research] 21.1:43-51.

———. 1999. “PROJECT 1 Revisited: On the Analysis and Interpretation of PR1 Tables.” In Otto

Kollath, Z. & Keuler, J. O. 2005. "Stellar acoustics as input for music composition". Musicae Scientiae, Special issue 2005-2006, 161-183.

Kopec, G. 1992. "Signal Representations for Numerical Processing," in A. Oppenheim et al. (Eds.). Symbolic and Knowledge-Based Signal Processing. Englewood Cliffs: Prentice Hall.

Kramer, Gregory, ed. 1994. Auditory Display: Sonification, Audification, and Auditory Interfaces. Santa Fe Institute Studies in the Sciences of Complexity. Procedings Volume XVIII. Reading, MA: Addison-Wesley. ISBN 0201626039.

Kramer, G., Walker, B. N., Bonebright, T., Cook, P., Flowers, J., Miner, N., et al. 1999. The Sonification Report: Status of the Field and Research Agenda. Report prepared for the National Science Foundation by members of the International Community for Auditory Display. Santa Fe, NM: International Community for Auditory Display (ICAD)

Kugel, P. 1990. “Myhill’s Thesis: There’s More than Computing in Musical Thinking.” Computer Music Journal 14(3): 12-25.

Langton, C. G. 1991. Life at the Edge of Chaos. In Artificial Life II, Proceedings Vol. X. SFI Studies in the Sciences of Complexity, Addison-Wesley.

Laske, O. 1973a. “In Search of a Generative Grammar for Music.” Perspectives of New Music 12(1): 351-378.

———. 1973b. “Toward a Musical Intelligence System: OBSERVER.” Numus West 4: 11-16.

———. 1980. "On Composition Theory as a Theory of Self-Reference," Allos. La Jolla: Lingua Press.

———. 1981. “Composition Theory in Koenig’s Project One and Project Two.” Computer Music Journal 5(4), In The Music Machine. Edited by Curtis Roads. Cambridge: MIT Press.

———. 1988. “Introduction to Cognitive Musicology.” Computer Music Journal 12(1): 43-57.

———. 1989. “Composition Theory: An Enrichment of Music Theory.” Interface 18(1-2): 45-59.

———. 1990. “The Computer as the Artist’s Alter Ego.” Leonardo 23(1): 53-66.

———. 1991. “Toward an Epistemology of Composition.” Interface 20(3-4): 235-269.

———. 1992a. “Artificial Intelligence and Music: A Cornerstone of Cognitive Musicology.” In Understanding Music with AI: Perspectives on Music Cognition. M. Balaban, K. Ebcioglu and O. E. Laske, eds. Cambridge: AAAI Press / MIT Press. 3-28.

———. 1993. “What is Composition Theory?.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 28-30.

———. 1999. Navigating New Musical Horizons. Westport: Greenwood Press

Laurson, M. and M. Kuuskankare. 2002. “PWGL: A Novel Visual Language based on Common Lisp, CLOS, and OpenGL.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 142-145.

Le Brun, M. 1979. "Digital Waveshaping Synthesis", Journal of the Audio Engineering Society, 27:4, p. 250

Leach, J. and J. Fitch. 1995. “Nature, Music, and Algorithmic Composition.” Computer Music Journal 19(2): 23-33.

Lerdahl, F. and R. Jackendoff. 1983. A Generative Theory ofTonal Music. Cambridge: MIT Press.

Little, D. 1993. “Composing with Chaos: Applications of a New Science for Music.” Interface 22(1): 23-51.

Lorrain, Denis. 1989. "A Panoply of Stochastic 'Cannons'." In The Music Machine. Edited by Curtis Roads. Cambridge: MIT Press.

Loy, D. G. 1989. “Composing with Computers: a Survey of Some Compositional Formalisms and Music Programming Languages.” In Current Directions in Computer Music Research. M. V. Mathews and J. R. Pierce, eds. Cambridge: MIT Press. 291-396.

Loy, D. G. and C. Abbott. 1985. “Programming Languages for Computer Music Synthesis, Performance, and Composition.” ACM Computing Surveys 17(2).

Lourenco, B.F., J.C.L.Ralha, M.C.P.Brandao. 2009. "L-Systems, Scores, and Evolutionary Techniques", Proceedings of 6th Sound and Music Computing Conference, Portugal, pp. 113-118

Luque, S. 2006. Stochastic Synthesis: Origins and Extensions. Master's Thesis, Institute of Sonology, Royal Conservatory, The Netherlands.

———. 2009. The Stochastic Synthesis of Iannis Xenakis. Leonardo Music Journal 19: 77-84

Manzolli, J., F. Damiani, P. J. Tatsch, and A. Maia. 2000. A Non-Linear Sound Synthesis Method. In Proceedings of the 7th Brazilian Symposium on Computer Music, Curitiba.

MacGregor, B. 2002. “Cybernetic serendipity revisited.” In Proceedings ofthe 4th conference on creativity & cognition. New York: ACM Press. 11-13.

Mackenzie, J.P. 1995. 'Chaotic Predictive Modelling of Sound'. Procs. International Computer Music Conference (ICMC '95), pp 49-56, Banff, Canada, September 1995.

MacLennan., B. J. 1990. “Continuous spatial automata”. Technical Report CS-90-121, University of Tennessee, Dept. of Computer Science, Knoxville. http://www.cs.utk.edu/mclennan/eldcompbiblio.html, accessed 04/01/2010.

Mandelbrot, B. 1982. The Fractal Geometry of Nature. New York: W. H. Freeman.

Manousakis, S. 2006. "Musical L-systems", Master's Thesis, Institute of Sonology, 2006

———. 2009. "Non-standard Sound Synthesis with L-systems", Leonardo Music Journal, MIT Press, issue 19, December 2009

Marino, G. and M. Serra, J. Raczinski. 1993. “The UPIC System: Origins and Innovations.” Perspectives of New Music 31(1): 258-269.

Marino, G. 1990. “The New UPIC System.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 249-252.

Mathews, M. V. 1963. “The Digital Computer as a Musical Instrument.” Science 142(3592)

———. 1969. The Technology of Computer Music. Cambridge: MIT Press.

Matossian, N. 1986. Xenakis. London: Kahn & Averill. McAlpine, K. and E.

Maurer, J.A., IV. 1999. ‘A Brief History of Algorithmic Composition’ Stanford University, accessed at: http://ccrma-www.stanford.edu/~blackrse/ algorithm.html (Dec 2004)

Mauss, Marcel. 1966. The gift; forms and functions of exchange in archaic societies. London: Cohen & West.

McAlpine, Kenneth. 1999. "Applications of Dynamical Systems to Music Composition." Ph.D. dissertation, Department of Mathematics, University of Glassgow.

McCartney, J. 1996. “SuperCollider: a New Real Time Synthesis Language.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association.

———. 2002. “Rethinking the Computer Music Language.” Computer Music Journal 26(4): 61- 68.

McCormack, J. 1996. “Grammar-Based Music Composition”. In Stocker et al, eds. Complex Systems 96: from local interactions to global phenomena, 321-336. IOS Press

McCormack, J. 2003. The Application of L-Systems and Developmental Models to Computer Art, Animation, and Music Synthesis. Ph.D. thesis, School of Com- puter Science and Software Engineering, Monash University, Clayton

McCormack, J. 2004. Impossible Nature: The Art of Jon McCormack. Melbourne: Australian Centre for the Moving Image.

McNabb, M. 1990. "The Far and Brilliant Night", inernet: http://www.mcnabb.com/music/works/fbn.html

McAlpine, K. and E. Miranda, S. Hoggar. 1999. “Making Music with Algorithms: A Case-Study.” Computer Music Journal 23(2): 19-30.

Miner, N. E., and Caudell, T. P. 2002. Using wavelets to synthesize stochastic-based sounds for immersive virtual environments. Presence: Teleoperators and Virtual Environments 11(5): 493–507.

Miranda, E. R. 1993. “Cellular Automata Music: An Interdisciplinary Project.” Interface 22: 3- 21.

———. 2000a. Composing Music With Computers. Burlington: Focal Press.

———. 2000b. “Regarding Music, Machines, Intelligence and the Brain: An Introduction to Music and AI.” In Readings in Music and Artificial Intelligence. E. R. Miranda, ed. Amsterdam: Harwood Academic Publishers. 1-13.

———. 2001. "Evolving Cellular Automata Music: From Sound Synthesis to Composition", Proceedings of the Workshop on Artificial Life Models for Musical Applications - ECAL 2001, Prague, Czech Republic.

———. 2002. Computer Sound Design: Synthesis Techniques and Programming. Oxford: Elsevier.

Miranda, S. Hoggar. 1999. “Making Music with Algorithms: A Case-Study.” Computer Music Journal 23(2): 19-30.

Miranda, E.R., McAlpine, K., Hoggar, S. 1997. Dynamical systems and applications to music composition: A research report. In: Proceedings of Journees d’Informatique Musicale (JIM97). French Society for Musical Informatics (SFIM), Lyon, France

Miranda, E. R. and Maia Jr., A. 2005. "Granular Synthesis of Sounds Through Markov Chains with Fuzzy Control", Proceedings of the International Computer Music Conference 2005, Barcelona

Miranda, E-R., Biles, J. (Eds.) . 2007. Evolutionary Computer Music. London: Springer

Monro, G. 1995. “Fractal Interpolation Waveforms”, Computer Music Journal, Vol. 19, No. 1, pp. 88-98

Moore, F. R. 1977 "Table Lookup Noise for Sinusoidal Digital Oscillators," Computer Music Journal 1(2) 26-29

———. 1980. “The Futures of Music.” Perspectives of New Music 19(1-2): 212-226.

———. 1990 Elements of computer music. Englewood Cliffs, N.J.: Prentice Hall.

Moorer, J. 1972. “Music and Computer Composition.” Communications ofthe ACM 15(2): 104-

Morgan, N. 2007. "Transformation and mapping of L-Systems data in the composition of a large-scale instrumental work", Proceedings of ECAL 2007 Workshop on Music and Artificial Life (MusicAL 2007), Lisbon (Portugal).

Munakata, N., and K. Hayashi. 1995. a Gene Music: Tonal AssignmentsofBasesandAminoAcids.o InProceedings of the 1995 Visualizing Biological Inform ation, Conference. Singapore: World Scientific, pp. 72-83.

Murail, T. 2005. The revolution of complex sounds. Contemporary Music Review, 24 (2).

Myhill, J. 1952. “Some Philosophical Implications of Mathematical Logic: Three Classes of Ideas.” Review of Metaphysics 6(2): 165-198.

———. 1978. “Some Simplifications and Improvements in the Stochastic Music Program.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 272-317.

Nattiez, J-J. 1990. Music and Discourse: Toward a Semiology of Music . Translated by Carolyn Abbate. Princeton: Princeton University Press.

Negroponte, N. 1995. Being Digital. London: Hodder & Stouton.

Nelson, G-L. 1993. "Sonomorphs: An Application of Genetic Algorithms to the Growth and Development of Musical Organisms." Proceedings of the Fourth Biennial Art And Technology Symposium.

———. 1994. "Wind, Sand, and Sea Voyages: An Application of Granular Synthesis and Chaos to Musical Composition." Available on-line at http://www-ks.rus.uni-stuttgart.de/people/schulz/fmusic/gnelson.html

———. 1996. Real Time Transformation of Musical Material with Fractal Algorithms. Computers Math. Applic. 32,1 (1996) 109-116.

Nierhaus,G. 2008. Algorithmic Composition - Paradigms of Automated Music Generation. Springer

Novikoff, A. 1945. "The Concept of Integrative Levels and Biology", Science, Volume 101, Issue 2618, pp. 209-215

Olson, H. F. and H. Belar. 1961. “Aid to Music Composition Employing a Random. Probability System.” Journal of the Acoustical Society of America 33(9): 1163-1170.

Oxford Advanced Learner's Dictionary 2006

Oxford Dictionaries Online. Accessed: http://oxforddictionaries.com/ (21/3/11)

Papadopoulos, George and G. Wiggins. 1999. "AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects." http://www.soi.city.ac.uk/ ~geraint/ papers/ AISB99b.pdf

Pinkerton, R. C. 1956. “Information Theory and Melody.” Scientific American 194(2): 77-86.

Peitgen, H-O., and D. Saupe, eds. 1988. The Science of Fractal Images. New York: Springer-Verlag.

Peitgen, H-O, H.Jurgens, and Saupe, D. 1992; Chaos and Fractals, New Frontiers of Science. New York, NY: Springer-Verlag.

Pereverzev, S. V., et al. 1997. aQuantum Oscillations between Two Weakly Coupled Reservoirs of Super- fluid 3He.o Nature 388:449± 451.

Polansky, L. and D. Rosenboom. 1985. “HMSL.” In Proceedings ofthe International Computer Music Conference. San Francisco: International Computer Music Association. 243-250.

Polotti, P. & Evangelista, G. 2001. “Fractal Additive Synthesis via Harmonic-Band Wavelets”. Computer Music Journal, 25:3, pp. 22–37, Fall 2001

Pope, S. T. 1995. “Fifteen Years of Computer Assisted Composition.” Proceedings of the Second Brazilian Symposium on Computer Music 6.

Pressing, J. 1988. “Nonlinear maps as generators of musical design”, Computer Music Journal, 12(2): 35-46.

Prusinkiewicz P. and A.Lindenmayer 1990. “The Algorithmic Beauty of Plants.” New York: Springer-Verlag,

———. 1994. “Novelty, Progress and Research Method in Computer Music Composition.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 27-30.

Puckette, M. 1991. “Combining Event and Signal Processing in the MAX Graphical Programming Environment.” Computer Music Journal 15(3): 68-77.

———. 1997. “Pure Data.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 224-227.

———. 2002. “Max at 17.” Computer Music Journal 26(4): 31-43.

———. 2007. The theory and technique of electronic music. World Scientific. ISBN 9789812700773.

Quinn, M and Meeker, L-D. 2001. "Research set to Music: The Climate Symphony and Other Sonifications of Ice Core, Radar, DNA, Seismic and Solar Wind Data". Proceedings of the 2001 International Conference on Auditory Display, Espoo, Finland

Rahn, J. 1990. “The Lisp Kernel: A Portable Software Environment for Composition.”, Computer Music Journal 14(4): 42-64.

Reichardt, J. 1968. Cybernetic Serendipity: The computer and the arts. London: Studio International, W. & J. Mackay.

Reiners, P. 2004. “Cellular automata and music”, developerWorks, IBM, 2004

Reynolds, S. 1996. Low end theory. http://www.mille-plateaux.net/theory/download/raynolds-thewire.pdf. As originally published in The Wire #196 (March 1996).

Risset, J-C. 1969. An Introductory Catalogue of Computer Synthesized Sounds, Bell Telephone Laboratories, Murray Hill, New Jersey, reprinted in The Historical CD of Digital Sound Synthesis, Computer Music Currents 13, Wer 2033-2, Schott Wergo Music Media GmbH, Meinz, Germany.

Roads, C. 1977. “Composing Grammars.” In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association. 26-30.

———. 1984. “An Overview of Music Representations.” In Musical Grammars and Computer Analysis. Firenze: Leo S. Olschki. 7-37.

———. 1985a. “Research in music and artificial intelligence.” In ACM Computing Surveys. New York: ACM Press. 17(2): 163-190.

———. 1985b "Interview with Gottfried Michael Koenig". In Curtis Roads and John Strawn, editors, Foundations of Computer Music, pages 568–580. MIT Press, Cambridge, MA, London, 1985.

———. 1996. The Computer Music Tutorial. Cambridge: MIT Press.

———. 2002. Microsound. Cambridge: MIT Press.

Roads, C. and Wieneke, P. 1979. Grammars as Representations for Music. Computer Music Journal. 3(1) (March 1979), 48-55

Robindoré, B. 1996. “Eskhaté Ereuna: Extending the Limits of Musical Thought-Comments On and By Iannis Xenakis.” Computer Music Journal 20(4): 11-16

Rodet, X. and P. Cointe. 1984. “FORMES: Composition and Scheduling of Processes.” Computer Music Journal 8(3): 32-48.

Rodet, X. 1993. "Models of musical instruments from Chua’s circuit with time delay", IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 40, no. 10, 696–701.

Russcol, H. 1972. The Liberation of Sound: An Introduction to Electronic Music. London: Prentice- Hall International.

Sangild, T. 2002. The Aesthetics of Noise. Datanom.

Scaletti, C. 2002. “Computer Music Languages, Kyma, and the Future.” Computer Music Journal 26(4): 69-82.

Schaeffer, P. 1970. “Music and Computers.” In Music and Technology (Proceedings ofthe Stockholm Meeting organized by UNESCO). Paris: La Revue Musicale. 57-92.

Schoenberg, Arnold. 1975. Style and Idea, edited by Leonard Stein with translations by Leo Black. Berkeley & Los Angeles: University of California Press.

Serra, M.H. 1993. “Stochastic Composition and Stochastic Timbre: GENDY3 by Iannis Xenakis.”. Perspectives of New Music 31, no. 1: 236-257

Serquera, J. Miranda, E. R. 2008. “Spectral Synthesis and Control with Cellular Automata” Proceedings of the International Computer Music Conference, Belfast, NI, 2008.

Shannon, C. E. 1948. “A Mathematical Theory of Communication.” Bell Systems Technical Journal 27: 379-423, 623-656.

Shannon, C. E. and W. Weaver. 1949. A Mathematical Theory of Communication. Urbana: University of Illinois Press.

Simoni, M. 2003. Algorithmic Composition: A Gentle Introduction to Music Composition Using Common LISP and Common Music. Ann Arbor: Scholarly Publishing Office, the University of Michigan University Library.

Smalley, D. 1986. “Spectro-Morphology and Structuring Processes.” In The Language of Electroacoustic Music. S. Emmerson, ed. London: Macmillan Press. 61-93.

Smalley, D. 1997. Spectromorphology: Explaining sound-shapes, Organised Sound: Vol. 2, no. 2. Cambridge: Cambridge University Press: 107-126.

Smith, J-O. 1991. ``Viewpoints on the History of Digital Synthesis", Proceedings of the International Computer Music Conference (ICMC-91, Montreal), pp. 1-10, Computer Music Association, October 1991.

———. 2005. Physical Audio Signal Processing. Draft. https://ccrma.stanford.edu/~jos/pasp05/

Smith, L. 1972. “SCORE — A Musician’s Approach to Computer Music.” Journal of the Audio Engineering Society 20(1): 7-14.

Soddell, F and Soddell, J. 2005. "Of Lindenmayer systems, Fungi and Music". Proceedings Australasian Computer Music Conference

Solomos, M. 2001. “The Unity of Xenakis’s Instrumental and Electroacoustic Music: The Case for ‘Brownian Movements’”. Perspectives of New Music 39, no. 1: 244-254

Spiegel, L. 1986. “Music Mouse — An Intelligent Instrument.” Internet: http://retiary.org/ls/programs.html.

———. 1989. “Distinguishing Random, Algorithmic, and Intelligent Music.” Internet: http://retiary.org/ls/writings/alg_comp_ltr_to_cem.html.

Stair, Ralph M., et al. (2003). Principles of Information Systems, Sixth Edition. Thomson Learning, Inc..

Stiny, G. and J. Gips. 1972. “Shape Grammars and the Generative Specification of Painting and Sculpture.” In Information Processing 71. C. V. Freiman, ed. Amsterdam: North Holland. 1460-1465.

———. 1978. Algorithmic Aesthetics. Berkeley: University of California Press.

Stone, Harold S. (1972). Introduction to Computer Organization and Data Structures (1972 ed.)

Sturm, B. L. 2001. ‘‘Composing for an Ensemble of Atoms: The Metamorphosis of Scienti□ c Experiment Into Music.’’ Organised Sound 6(2):131–145.

Supper, M. 1997: Elektroakustische Musik & Computermusik, Wolke Verlag: Hofheim

———. 2001. “A Few Remarks on Algorithmic Composition.” Computer Music Journal 25(1): 48-53.

Tache, O. and Claude Cadoz (2009). Organizing mass-interaction physical models: the CORDIS-ANIMA Musical Instrumentarium. Proceedings of the 2009 International Computer Music Conference, Montréal (Canada), pp. 411-414.

Taube, H. 1997. “An Introduction to Common Music.” Computer Music Journal 21(1): 29-34.

———. 2004. Notes from the Metalevel. New York: Routledge.

Tenney, J. 1963. “Sound Generation by Means of a Digital Computer.” Journal of Music Theory 7(1): 24-70.

———. 1966. “Musical Composition with the Computer (Abstract from the 71st Meeting of the Acoustical Society of America).” Journal of the Acoustical Society of America 39(6): 1245.

———. 1969. “Computer Music Experiments.” In Electronic Music Report. Utrecht: Institute of Sonology. 1: 23-60.

———. (1988). Stochastic String Quartet. Score, Smith Publications/Sonic Art Editions, Baltimore, USA.

Thomson, P. 2004. "Atoms and errors: towards a history and aesthetics of microsound". Organised Sound, 9(2).

Toguchi, S., Akamine, Y., Satoshi, E.,2008. “Research into the Generation of Sound Effects Using a Cellular Automaton” Lecture Notes in Computer Science: Cellular Automata, Volume 5191/2009, Springer.

Truax, B. 1973. “The Computer Composition — Sound Synthesis Programs POD4, POD5 and POD6.” In Sonological Reports. Utrecht: Institute of Sonology. 2: 57.

———. 1976. “A Communicational Approach to Computer Sound Programs.” Journal of Music Theory 20(2): 227-300.

———. 1982. Timbral Construction in Arras as a Stochastic Process", Computer Music Journal, 6(3), 1982

———. 1990. “Chaotic Non-Linear Systems and Digital Synthesis: An Exploratory Study”. Proceedings of ICMC, Glasgow, pp100-103, International Computer Music Association, San Francisco, 1990.

———. 1992. "Musical Creativity and Complexity at the Threshold of the 21st Century," Interface, 21(1), 1992, 29-42.

———. 1997. "The Inner and Outer Complexity of Music," Perspectives of New Music, 32(1), 1994, 176-193. Italian translation in Musica/Realtà , No. 43, April 1994; Polish translation in Monochord, 14-15, 1997.

———. 1999. "Sonology: A Questionable Science Revisited", In Tabor, J. (Ed.) Otto Laske: Navigating New Musical Horizons. London: Greenwood: 21-36.

———. 2000. "The aesthetics of computer music: a questionable concept reconsidered," Organised Sound, 5(3), 119-126, 2000

———. 2001. Acoustic Communication, 2nd ed., Ablex Publishing.

Vaggione, H. 2001. “Some Ontological Remarks about Music Composition Processes.” Computer Music Journal 25(1): 54-61.

Varèse, E. 2004. “The Liberation of Sound.” In Audio Culture: Readings in Modern Music. C. Cox and D. Warner, eds. New York: Continuum. 17-21.

Varga, B-A. 1996. Conversations with Iannis Xenakis. London: Faber and Faber.

Voss, R. F. and J. Clarke. 1975. “1/f Noise in Music and Speech.” Nature 258: 317-318.

Vickers, P. "Ars Informatica - Ars Electronica: Improving Sonification Aesthetics," in Understanding and Designing for Aesthetic Experience Workshop at HCI 2005 The 19th British HCI Group Annual Conference (L. Ciolfi, M. Cooke, O. Bertelsen, and L. Bannon, eds.), (Edinburgh, Scotland), 2005.

Wallin, R. 1989. "Fractal Music - Red Herring or Promised Land? or Just Another of those Boring Papers on Chaos", Lecture given at the Nordic Symposium for Computer Assisted Composition Stockholm 1989. Internet: http://www.rolfwallin.org/Fractalarticle.html

Waschka, R., Kurepa, A. 1989. “Using Fractals in Timbre Construction: an Exploratory Study”. Proceedings of ICMC, Columbus, pp332-335, International Computer Music Association, San Francisco, 1989.

Weisstein, E.W. 2006. "Bernoulli Distribution." MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/BernoulliDistribution.html

Whitelaw, M. 2003. "Sound Particles and Microsonic Materialism", Contemporary Music Review: Vol. 22, Issue 4. London: Routledge: 93-100.

Whitelaw, M. 2004. Metacreation: Art and Artificial Life. Cambridge, MA: MIT Press

Whittall, A. 2008. The Cambridge Introduction to Serialism. Cambridge Introductions to Music, p. 272. New York: Cambridge University Press.

Wiener, N. 1948. Cybernetics. Cambridge: MIT Press.

Wilson, G. 1988. “The Life and Times of Cellular Automata”, New Scientist October 1988:44-47, Reed Business Information.

Wilson, L. B. 1993. Comparative Programming Languages, Second Edition. Addison-Wesley.

Winograd, T. 1979. “Beyond Programming Languages.” Communications of the ACM 22(7): 391-401.

Wishart, T. 1994. Audible design. OTP Ltd.

Wolfram, S. 1983. Statistical Mechanics of Cellular Automata. Reviews of Modern Physics, 55(3): 601-644

———. 1984. Universality and Complexity in Cellular Automata. Physica D. 10D, 1-35.

———. 2002. A New Kind of Science. Wolfram Media.

Wooller, Rene, Brown, Andrew R, Miranda, Eduardo, Diederich, Joachim, & Berry, Rodney (2005) A framework for comparison of process in algorithmic music systems. In David, Burraston & Ernest, Edmonds (Eds.) Generative Arts Practice, 5-7 December 2005, Sydney, Australia.

Worth, P., S. Stepney. 2005. "Growing Music: Musical Interpretations of L-Systems". EvoWorkshops: 545-550

Xenakis, I. 1955. “La crise de la musique sèrielle.” Gravesaner Blatter 1.

———. 1965. “Free Stochastic Music from the Computer. Programme of Stochastic music in Fortran.” Gravesaner Blä tter 26.

———. 1966. “The Origins of Stochastic Music.” Tempo 78: 9-12.

———. 1976. Foreword. N’Shima [score]. Paris: Editions Salabert.

———. 1985. “Music Composition Treks.” In Composers and the Computer. C. Roads, ed. Los Altos: William Kaufmann, Inc.

———. 1992. Formalized Music: Thought and Mathematics in Music. Indiana: Indiana University Press.

———. 1996. "Determinacy and Indeterminacy." Organised Sound 1(3): 143-155.

Yeh, D. T. and Pakarinen, Jyri (2009). "A Review of Digital Techniques for Modeling Vacuum-Tube Guitar Amplifiers", Computer Music Journal, 33:2, pp. 89-90

Yadegari.Sh. D. 1991. “Using Self-Similarity for Sound-Music Synthesis.” International Computer Music Conference 1991.

———. 1992. “Self-Similar Synthesis - On the Border Between Sound and Music.” MIT Thesis.

Zalta, N. 2011. Stanford Encyclopedia of Philosophy. Accessed (13/8/2011): http://plato.stanford.edu/

Zaripov, R. 1969. “Cybernetics and Music.” Perspectives of New Music 7(2): 115-154.

Zicarelli, D. 1987. “M and Jam Factory.” Computer Music Journal 11(4): 13-29.