# Realtime Algorithmic Music Systems From Fractals and Chaotic Functions: Toward an Active Musical Instrument

by

Rubén Hinojosa Chapel

Submitted in partial fulfilment of the requeriments for
the degree of Diploma of Advanced Studies
Doctorate in Computer Science and Digital Communication
Department of Technology

Tutor: Dr. Francesc Xavier Serra Casals

Universitat Pompeu Fabra
Barcelona, September 2003

# Abstract

Due to high degree of formalism that music has on its theoretical aspects, composers have developed throughout centuries of musical practice, many compositional methods that actually are algorithmic procedures. Mathematical thinking has been an essential component of our western music. That is why since computers were born, composers have been seduced by its abilities for handling mathematical structures. From a certain point of view, a music score is a mathematical structure.

My research work focuses on a particular approach to symbolic modeling of music, concerned with the possible development of a new model of computer music system: Active Musical Instruments. I will show that the basis of this model is present in several ideas that have emerged in the recent past. I will also show that here converge concepts from relevant research areas that are under continuous and growing development, such as: Algorithmic Music Systems, Interactive Music Systems, Mapping, and Human Computer Interfaces.

I intend to contribute to the setting-up of a formalized framework for the study of Active Instruments as a model of a new kind of computer-based musical instrument. So, the elaboration of a theoretical, conceptual and methodological framework for the study, design and development of these instruments will be proposed.

Active instruments own a generative nature, which imply a generative model for its core. I will explain why iterated and chaotic functions (Ifs) could be a good candidate for the development of this core. Thus, I will also propose the study of this kind of model inside the proposed framework as a particular and narrow approach to the nature of the problem. In fact, in this research work I propose the development of the following relation:

**Active Instrument = Realtime Algorithmic Music (Ifs?) + Interactive Control**

Finally, I will suggest some possible future directions for achieving my objectives.

# Acknowledgments

I would like to thank the following people:

- Composers Carlos Fariñas, Roberto Valera and Fernando Rodríguez (Archie), and all the students of Music Composition at the *Facultad de Música* of the *Instituto Superior de Arte* in Havana, for their contribution with ideas, suggestions, comments, discussions, critics and direct experiences with the projects I was working on during my time as researcher and professor in that place (1990-2000).

- Carlos A. Gonzalez Denis for having introduced me to fractal theory.

- Composers John Bischoff and Chris Brown for their bibliographic contribution.

- Composer Gabriel Brncic for his encouragement and support for making my Doctor degree at the *Universitat Pompeu Fabra* in Barcelona.

- Composer Andrés Lewin Richter for using one of my algorithmic music composition programs for creating a composition.

- Sergi Jordà for his valuable discussions.

- Dr. Xavier Serra for his support to my stay in the Music Technology Group and the Ph.D. program at the *Universitat Pompeu Fabra*, and for his valuable suggestions as advisor.

*La música, alianza de la magia algebraica con las habilidades y los cálculos de la armonía, en constante y atrevida guerra contra la razón y la sobriedad.*

Thomas Mann
Doktor Faustus

*Schönberg –ha dicho alguien– es un inventor de teoremas, que siempre parece estarlos demostrando en el encerado, con fórmulas geométricas... Lo cierto es que la música no puede nunca recurrir a las fuerzas de lo inconsciente, como la poesía o la pintura; es un arte –aunque me atemorice el término– terriblemente euclidiano. Y una fuga de Bach, como un motete de Victoria, pueden siempre traducirse en el encerado por medio de fórmulas geométricas.*

Alejo Carpentier
Crónicas

***Computer Music** is neither a style nor a genre; it is simply music for which the use of a computer is necessary -or at least central- to its genesis. This encompasses computation of electro-acoustic sonic material as well as the computation of a score. Computer music can be instrumental, vocal or electro-acoustic. Generating musical scores by computer is generally known as algorithmic composition or computer-aided composition, but also encompasses a special class: "interactive composing" is a term that was introduced in 1967 by Joel Chadabe (...). Chadabe describes interactive composition as composing with a real-time system during a performance.*

ICMC 2000
Call for Works

# Contents

# List of Figures and Tables

# Chapter 1

# Introduction

This is a multidisciplinary research work, presented as a research retrospective. It is the result of two wide stages of research. The first period took place between 1990 and 2000 in Havana, Cuba, inside the EMEC / ISA (*Estudio de Música Electroacústica por Computadora* / *Instituto Superior de Arte*). The second one starts in the year 2001, and continues just to the present days. It is being carried out in the frame of the Ph.D. programme in Computer Science and Digital Communication of the *Universitat Pompeu Fabra*, in Barcelona, Spain, inside the MTG (Music Technology Group). Though the general field encompassed by this paper is Algorithmic Music Composition Systems, or simply Algorithmic Music, it is largely concerned with a particular, novelty and an up-to-date subfield: algorithmic music composition in realtime; i.e., music composition by means of algorithms implemented in a computer system, played live, in an interactive way, in front of the audience, immersed in a configuration which defines a kind of instrument usually classified as *intelligent*.

## 1.1 Some relevant definitions

1- When the user introduces initial data (*seed data*) into the program, waits for a while, and gets and assesses the results, we say we have a **non-realtime** algorithmic music system.
2- When the generation of musical material takes place live, we say we have a **realtime** algorithmic music system. This system could be **interactive** or **non-interactive**.
3- When the user can influence the sonic behaviour of a realtime algorithmic music system, while simultaneously listens to the generated musical material, we say we have a **realtime interactive** algorithmic music system. This kind of system could be the core of a class of computer-based instruments I prefer to name *Actives.*

## 1.2 The nature of the problem

Interactive composing pioneer Joel Chadabe states that generatives algorithms are "*the engine of a new type of musical instrument*". The concept of the kind of instrument he describes, puts the notion of algorithmic generation of musical materials into an interactive context. "*Not only does the instrument **generate** information automatically according to some algorithm, it allows the performer / composer to **guide** the activity of the algorithm, thereby maintaining a certain control of the details in the resulting work of art*" (Chadabe, 2001) (bold is our).

A realtime interactive algorithmic music system owns a generative nature. To my mind, its core could be built from almost any mathematical model. **Fractal** and **Chaos** theories are one of the many approaches to construct this core. In fact, in my research work I propose the development of the following relation, where *Ifs* stands for *Iterated Functions*:

**Active Instrument = Realtime Algorithmic Music (Ifs?) + Interactive Control**

I intend to contribute to the setting-up of a formalized framework for the study of Active Instruments, as a particular view of the conjunction (or intersection) of two research areas: **Algorithmic Music Systems** and **Interactive Music Systems**. In this research work converge concepts from both fields. For a better understanding of the nature of the problem, let's take a look at the Figure 1.1, which represents the proposed structure of an Active Musical Instrument:

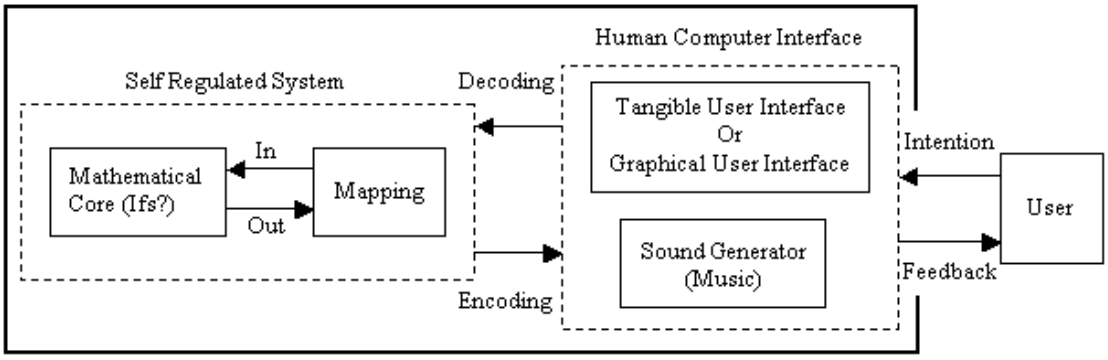



**Figure 1.1** Proposed structure of an Active Musical Instrument.

From the analysis of this diagram several questions arise:

**Questions that arise from the Mathematical Core**

1- Are Iterated Functions and Chaotics Systems well suited for building the Mathematical Core?
2- Which kind of Iterative Models are best suitable, these which result in simple images (atractors, logistic function), or the ones which generate complex images (Mandelbrot / Julia sets)?
3- Which are the pros and cons of each model?
4- Which approach is best suitable, the generation of musical material without the realtime graphic representation of the image, or the realization of such drawing?
5- Which are the pros and cons of each approach?
6- When using Iterative Models as generative algorithms for interactive realtime computer assisted composition, how can we achieve interactivity, low latency feedback and intuitivity? That is: how can we effectively influence the behaviour of the model? How can we perceive that behaviour with the minimum delay (immediate aural feedback), and therefore, how can we intuitively learn to control that behaviour? Is it possible to achieve the cause-and-effect aspect inherent of traditional musical performance?

**Questions that arise from the Mapping**

1- What can be considered as a good mapping? The one that actually reflects the mathematical behaviour of the core, without taking into account the musical results? The one that results in "satisfactory" musical outputs, without taking into account the actual mathematical behaviour of the core?
2- Are equally suitable simple and complex mapping? For us a complex mapping includes elements of musical knowledge (concepts, notions…) and could be very tricky, while simple mapping remains "unmusical" and algorithmically clear and direct.

**Questions that arise from the whole**

1- Is the term "Intelligent Instrument" appropiate for such a system? My proposition is that "Active Instrument" is a better one. In fact, the later term has already been used in the past, in some cases, as a synonym of the former.
2- How can we assess such a system? What requirements should be addressed in order to build a satisfactory system? Satisfactory for whom and what?
3- What is the role of such a system?
4- Are they musically expressive enough, just like traditional instruments are?

**Questions that arise from the Human Computer Interface**

1- What models of HCI can be designed for interactive realtime algorithmic composition / improvisation and performance? My proposition is that both GUI and TUI (Tangible User Interface) are well suitable.
2- What are the requirements of these models for better support creativity?
3- Are they intuitive enough? That is: does the interface make it easier for the user to establish a mental schema of the system he is controlling?
4- More about intuitivity: what design model is best suited for control, the one that exhibits common and/or musical knowledge and hides the mathematical aspects of the core, or the one that reflects the inside mathematics? Are both approaches valid and useful?
5- Are they flexible enough? That is: does the interface easily adapt for the needs and requirements of "divergent" and "convergent" thinking?

**Questions that arise from the Music**

1- Is the resulted algorithmic music an aesthetical meaningful form of Art?
2- What can be its "utility"?
3- How can we guarantee, to some extend, musical logic and coherence?

**Questions that arise from the User**

1- What is the role of the user/musician?

**Philosophical and ideological questions (concerned with algorithmic composition in general)**

1- Does it has sense to look for a definitive composition system?
2- Who is the author of the composed music: the user, the designer of the algorithm / system or the machine?
3- What is the utility of these systems? Why should we investigate such systems?
4- Why general people, even scientific researchers, are "skeptical" about these systems?

## 1.3 Objective of this research

The problem of investigate and study active instruments is multidisciplinar by nature. To my mind, it needs a holistic approach: scientific, technological, artistic and philosophical, because all these points of view are linked and influenced between themselves.

The objective of this research work is *to contribute to the field of interactive realtime algorithmic music systems by identifying and addressing the aforementioned questions*, building upon the hypothesis that new meaningful sonic results and instrument-performer relationships can be achieved by Active Instruments.

I intend to contribute to the setting-up of a formalized framework for the study of Active Instruments as a model of a new kind of computer-based musical instrument. Thus, *I aim at the development of a theoretical, conceptual and methodological framework for the study, design and development of these instruments.*

A specific and narrow approach is the development of generative algorithms from iterated and chaotics functions for building the mathematical core. Thus, *I aim at researching on the generation of interactive realtime symbolic music data resulting from the application of chaotic systems.*

From my point of view, this work could be useful in research areas that are taken into account and converge, such as algorithmic composition systems, interactive music systems, design of environments for supporting creativity, or whatever project intended to manipulate the features of music for artistic purposes.

## 1.4 Why am I interested in Ifs and chaotic systems?

Fractal images are generated by means of relatively simple calculi, repeated again and again, using recursively on each step the results of the previous step. In many cases they are calculated by means of *Iterated Functions*. The numbers generated by an iterated function exhibit three types of behavior: steady-state, periodic, and chaotic. I focus on the creation of musical material and its organization arising from the realtime computation of iterated functions which exhibit chaotic behaviour. Chaotic functions are special cases on nonlinear dynamic systems. They represent deterministic processes that are very sensitive to initial starting conditions. That is why the theory that studies such systems is often referred to as the chaos theory, which is:

> *The qualitative study of unstable aperiodic behavior in deterministic nonlinear dynamical systems.* (Kellert, 1993)

Thus, I am interested in these kind of formalisms because:

1- They change and develop in time, like music does. Music is a temporal process, while dynamic systems are mathematical models of temporal processes.
2- They are non-stochastic processes, that is: for the same initial configuration they always behave the same way through time. Thus, they are suitable for music

composition in a similar way traditional music does. We can "compose" previously a composition, that is, an initial configuration and a development plan, and later perform the same composition whenever we want. This assures we will perform always the same composition, just like in traditional music.

3- Additionally, when we interact in realtime, a degree of unpredictability is present. So, because time imprecisions of the human performer, all performance will slightly be diferent, just like when performing traditional music.
4- This degree of unpredictability is also useful and desirable for interactive improvisation / composition.
5- Due to the high sensitive dependence on initial conditions, when we make slight changes to a system's starting conditions, the later behavior of the system may soon become completely different. This is a good feature that enables the generation of a high amount of different musical outcome from the same chaotic system.
6- They exhibit a highly complex unstable aperiodic behavior; the variables describing the state of a system do not demonstrate a regular repetition of values. This feature is important for achieving *contrast*. In this case, the challenge is to "insert" *analogy* into the system. I seek both properties in the generated musical material in order to achieve, to some extend, musical logic and coherence.

As introduced before, a realtime interactive algorithmic music system owns a generative nature. To my mind, its generative core could be built from almost any mathematical model. The aforementioned six reasons explain why iterated chaotic functions (Ifs) seem to be a good candidate for the development of this generative core. The study of this kind of model inside the proposed framework gives us a particular and narrow approach to the problem of investigate and study active instruments. That is why I propose the study and development of the aforementioned relation:

**Active Instrument = Realtime Algorithmic Music (Ifs?) + Interactive Control**

"**(If?)**" means that iterated functions are one of the many approaches for building the generative core. Actually, for narrow this relation I have chosen Ifs. Thus, this equation will become as follows, without the question mark:

**Active Instrument = Realtime Algorithmic Music (Ifs) + Interactive Control**

## 1.5 Proposed requirements of this work

I intend to achieve the following requirements in this work:

1- Should be not-inspired (no mimic) on any known musical style. The use of general simple music knowledge is permitted, though.
2- Guarantee, to some extend, satisfactory aesthetic results, musical logic and coherence.
3- Bridge the gap between amateur and professional users.
4- Provide a reasonably short learning curve.

Researcher Rajmil Fischman states that "*no theory is capable of guaranteeing satisfactory aesthetic results or musical coherence*" (Fischman, 2003). Nevertheless, I am interested in algorithms which preserve musical logic and coherence. One of my aims is the musical coherence and interest of the sonic result. Thus, both theory and software are intended to guarantee, to some extend, satisfactory aesthetic results. A specific example is my electronic piece *El fin del Caos llega quietamente*, which is aimed at demonstrating partial progresses of our experiments. All notes were calculated by my *Fractal Composer* system from the *Logistic Function*, in realtime interaction with the author, and recorded in one pass with no overdubbing.

## 1.6 Context, research approach and methodology

The present work has been carried out in two stages. The first one took place between 1990-2000 in the frame of the EMEC/ISA, a Cuban artistic institution where I worked close and together with composers, in naturalistic situations, participating in their creative processes. Among other interests, the EMEC/ISA aims at scientific researching in computer science for artistic music applications. Apart from been scientifically valid, projects of the EMEC should also be musically relevant. During this first stage I collaborated with electroacoustic and acoustic music composers, in order to learn and provide our research in music modeling with a theoretical basis for the development of composition tools with their actual needs and requirements.

We made several experiments *toward the general study and investigation of algorithmic music*, focusing on the development of generative models for assistance in music composition (electroacoustic and instrumental music). I followed a bottom-up approach, going from simple and knew things (musical scales, melodic variations…) to complex and unexperimented ones, taking into account a holistic point of view. Work carried out consisted in the development of four projects, each one scheduled in the following four main steps:

1- Development of related theoretical principles and algorithms for the generation of music,
2- their implementation as a compositional software,
3- and their realization as one or more compositions that demonstrate their musical validity and viability, and the usefulness of the software.
4- An additional step was the analysis of previous ones, in order to get conclusions and propose future directions.

While satisfactory partial results were obtained, they were applied to the composition of artistic works of music, which have been played in Cuba and abroad in the frame of festivals, congresses and concerts. These results were also publicly presented in different scientific or artistic meetings, by means of concerts, papers and lectures. Part of this research work has been awarded national acknowledgements. It is important to point out that the carried work has always counted on the personal support and close collaboration from two reputed Cuban composers: Carlos Fariñas (1934-2002) and Roberto Valera. They were actually invaluable advisors, making critics, comments, suggestions… They have contributed their ideas, their broad musical experience and deep music knowledges. From the beginning they trusted in the creative possibilities of primary software versions for doing their compositional labour.

The second stage started in 2001, in the context of the Pompeu Fabra IUA-MTG (Institut Universitari de l'Audiovisual – Music Technology Group), and goes on up today. During this new period I learned complementary subjects in the frame of the Ph.D. programme in Computer Science and Digital Communication. This context has been a good environment:

1- for search and evaluate relevant information from the web;
2- for consult relevant scientific books and magazines;
3- for rethink and develop several ideas that emerged from the first stage;
4- for compare and contrast our research with another works;
5- for experiment new approaches to the nature of the problem;
6- and for formalize the argument I intend to develop through this dissertation, i.e. a framework for active instruments (based on chaotics functions).

Specifically important to this second stage have been our relationship and discussions with two experienced researchers on computer music: Sergi Jordà, chief of our Interactive Sonic Systems group, and Dr. Xavier Serra, my tutor and director of the MTG.

## 1.7 Structure of this document

In this introductory chapter, some relevant definitions for the understanding of the nature of the problem have been introduced. The problem and objective of this research, that is, the study and development of a computer-based instrument model, have been presented. Here I have explained why I am interested in iterated chaotic functions for building the generative core of active instruments, and thus, why this way we narrow the nature of the problem. Finally, the proposed requirements of this work, its context, research approach and methodology were exposed.

The structure of this document remains as follows:

Chapter 2 provides a review of some early projects where the main notions of Active Instruments are present, as well as a review of possible applications of fractals and chaos theories to music composition. My previous related work is presented in Chapter 3. The reader will find there the objectives and methodology of this work, as well as the experimental design and a detailed description of each carried out project. Chapter 4 presents a theoretical discussion around Active Instruments, while Chapter 5 outlines some possible future directions for achieving the objectives of this research work.

# Chapter 2

# Background

*In 1981, I was invited to deliver a keynote address at the International Music and Technology Conference in Melbourne, Australia. In preparing for that conference, I coined the term interactive composing to describe a performance process wherein a performer shares control of the music by interacting with a musical instrument.*

Joel Chadabe
Electric Sound

## 2.1 Introduction

Due to the mathematical formalism of music, it can be composed by systematically applying algorithmic procedures, even without computers. In fact, composers have developed throughout centuries of musical practice, compositional procedures which actually are algorithms, or algorithmic procedures. There are a lot of examples of precomputers algorithmic composition practices (Loy, 1988), such as the dice game developed by Mozart (*Musikalisches Würfelspiel*) or the stochastic music created by Iannis Xenakis starting in the fifties of the past century (*Metastasis*, 1955). The twelve-tone formalism developed almost a century ago by Arnold Schoenberg conforms indeed a set of algorithmic procedures.

With the advent of computers, many composers have been seduced by its abilities for handling complex structures, like music scores. Starting with the experiences carried out since the fifties by Hiller and Isaacson (Hiller, 1959, 1964, 1981), there is a long list of pioneers (Roads, 1996) such as: Iannis Xenakis (Xenakis, 1991), Herbert Brün, Gottfried Michael Koenig, John Myhill, Rudolf Jafizovich Zaripov (Zaripov, 1971), James Tenney, Pierre Barbaud, Michel Phillipot and Barry Truax. David Cope (Cope, 1991, 1996, 2000, 2001) is well known because his research and impressive results in composition style simulation since the past eighties. Gareth Loy‘s *Survey of Some Compositional Formalisms and Music Programming Languages* (Loy, 1988) is a good introduction to early approaches.

A central idea of an Active Instrument is the notion of a self-regulated sonic system, which owns a personal sonic behaviour, that can be controlled in realtime in an interactive way. This kind of system automatically proposes the composer musical materials live, in realtime, while he influences this sonic behaviour in an interactive way. In this chapter I will mention and review several hardware and software examples which exhibit this basic notion. From the observation of these examples plus the

analysis of my own related work, I found common notions that establish a relationship between all these projects. From my point of view, the existence of these common notions suggests that today there exists an underlying model of computer-based instrument that needs to be studied.

A realtime interactive algorithmic music system owns a generative nature. There are a lot of approaches for the generation of symbolic music data (Roads, 1989; Papadopoulo, 1999). To my mind, the mathematical core of an Active Instrument could be built around any generative algorithm. *Fractal* and *Chaos* theories are one of the many approaches to construct this core. Thus, I have chosen iterated and chaotic functions as a particular and narrow approach to the study of the aforementioned underlying model. We will see some applications of these theories to music composition.

## 2.2 Emerged ideas for an active instrument

The notion of self-regulated sonic system, which owns a personal sonic behaviour that can be controlled in realtime in an interactive way, is fundamental for the model of active instuments. This notion seems to have emerged since the fifties of the past century:

> *The use of modern digital tools for the construction of "active" (or even "intelligent") performance instruments has been underway since the appearance of the first analog music synthesizers in the 1950's.* (Rolnick, 1992)

Although with different approaches, the Barron's electronic circuits ('50s), the Raymond Scott's Electronium ('50s), the Max Mathews' GROOVE system ('70s), the Laurie Spiegel's software ('70s), the Salvatore Martirano's SalMar Construction ('70s), and the Joel Chadabe's CEMS system ('70s) share a similar idea. With the advent of personal computers in the eighties, appeared several comercial software that could be classified as active instruments. Among others there are: Dr. T's, Sequencer, Music Mouse, Instant Music and M. "Intelligent Instrument" was an often used term for classifying most of these projects. Let us review now these approaches.

### 2.2.1 The Barron's circuits

I find very interesting and pioneering the works done by Louis and Bebe Barron during the fifties of the past century. Influenced by Norbert Weiner and his work in cybernetics, the Barrons created their own circuitry and recorded the results on tape. They intended to build new sonic models using the spontaneous electric evolution of some electronic circuits coupled between themselves, whose oscillation frequencies were placed in the audible range. The main idea was to build series of active circuits with specific frequencies and transitory regime. By coupling these circuits to each other and influencing the behaviour of its neighbour circuits, it is possible to make changes to its own parameters. According to a partially predictable process, the union of synchronizing influences coming from its neighbour oscillators will modify the state of the oscillations of each circuit, so that they modulate their oscillations between themselves (Moles 1960).

The first circuit state is dictated by external conditions, which can be changed at will. Leaving it to itself, the system of circuits follows an evolutionary process, which can be defined as the behaviour in reaction to external stimuli. This acoustic behaviour is modified according to the relationship and order established between the circuits, and confers personal characteristics to a particular considered system (Moles 1960).

If we choose and study conveniently the parameters of those circuits, it could be possible to obtain an interesting sonic result, which could lead to the creation of an electronic music composition. Under this perspective, Louis and Bebe Barron made music for the cinema, especially for the films *Bells of Atlantis* (1952), *Electronic Jazz* and the science fiction film *Forbidden Planet* (1956). The soundtrack of this film is a wonderful example of artistic and avant-garde creation, and a remarkable example of the musical use of sound synthesis by modulation. It is considered as the first electronic score for a commercial film. The CD edition of the soundtrack contains the following revealing "music notes":

> *We design and construct electronic circuits which function electronically in a manner remarkably similar to the way that lower life-forms function psychologically. There is a comprehensive mathematical science explaining it, called "Cybernetics", which is concerned with the Control and Communication in the Animal and the Machine. It was first propounded by Prof. Norbert Wiener of M.I.T. who found that there are certain natural laws of behavior applicable alike to animals (including humans) and electronic machines.*
>
> *In scoring FORBIDDEN PLANET –as in all of our work– we created individual cybernetic circuits for particular themes and leit motifs, rather than using standard sound generators. Actually, each circuit has a characteristic activity pattern as well as a "voice". (...) There were no synthesizers or traditions of electronic music when we scored this film, and therefore we were free to explore "terra incognito" with all its surprises and adventures.* (Barron, 1989)

Although their circuitry was not a computer-based interactive musical instrument, properly speaking, in their work I find early notions of:

1- A self-regulated sound generation system
2- which owns a personal and autonomous sonic behaviour

### 2.2.2 The Electronium

In the fifties of the past century, American composer and great inventor Raymond Scott, designed and built the first of many very different versions of THE RAYMOND SCOTT **ELECTRONIUM**, a keyboard-less, automatic composition and performance machine programmed by knobs and switches. It should not be confused his Electronium with the Elektronium, a keyboard instrument invented by the Hohner company in 1950, and used by composers such as Stockhausen.

Building on the foundations of, and cannibalizing components from, previous music machines, his Karloff generator and Wall of Sound sequencer, Scott developed the first version of his "instantaneous composition/performance machine" in the late 1950s, and it became the most ambitious and resource-consuming project of his life. He wrote in a patent disclosure:

> *The entire system is based on the concept of Artistic Collaboration Between Man and Machine. The new structures being directed into the machine are unpredictable in their details, and hence the results are a kind of duet between the composer and the machine*. (Winner & Chusid)

Instead of a traditional, piano-style keyboard, the Electronium was guided by a complex series of buttons and switches, arranged in orderly rows. The system was capable of "instantaneous composition and performance" of polyphonic rhythmic structures, as well as tasking preset programs. With Scott controlling the sonorities, tempos, and timbres, he and his machine could compose, perform, and record all at once. The parts weren't multitracked; rather, voices, rhythms, and melodies originated simultaneously in real time. (Winner & Chusid)

The composer should request the Electronium to suggest a motive, which is then played through a loudspeaker. When the musician hears a satisfactory motive, he presses a switch and then, the relays and the drum memory of the machine are set in motion. The composer could modify the resulting music by means of knobs and switches, but in an unpredictable way. According to Scott, "*The Electronium is not played, it is guided"* (Roads, 1996). He wrote in the user manual:

> *A composer 'asks' the Electronium to 'suggest' an idea, theme, or motive. To repeat it, but in a higher key, he pushes the appropriate button. Whatever the composer needs: faster, slower, a new rhythm design, a hold, a pause, a second theme, variation, an extension, elongation, diminution, counterpoint, a change of phrasing, an ornament, ad infinitum. It is capable of a seemingly inexhaustible palette of musical sounds and colors, rhythms, and harmonies. Whatever the composer requests, the Electronium accepts and acts out his directions. The Electronium adds to the composer's thoughts, and a duet relationship is set up*. (Winner & Chusid)

### 2.2.3 The GROOVE system

In the late 1960s and early 1970s, Max V. Mathews and F. Richard Moore developed a system for realtime control of analog synthesizers called GROOVE, at AT&T Bell Telephone Labs. On Mathews's own words: "*Starting with the Groove program in 1970, my interests have focused on live performance and what a computer can do to aid a performer*" (Spiegel's webpage). According to Laurie Spiegel's view, who worked with Mathews in the seventies, the system was ideal "*for the development of what we called "intelligent instruments". (...) It also made the system ideal for the exploration of compositional algorithms.*" (Spiegel, 1998a). She gives a good description of the GROOVE system:

> *Before going on to its visual applications, it may help to help you visualize the GROOVE system in its original form, that of a hybrid (digital-analogue) computer music system, as developed by Max Mathews, Dick Moore and colleagues. The principle was both simple and general. A number of input devices (knobs, pushbuttons, a small organ keyboard, a 3D joystick, an alphanumeric keyboard, card reader, several console and toggle switches, and a number of output devices (14 digital-to-analog converters used for control voltages, 36 computer controlled relays, a thermal printer, and 2 washing machine sized one megabyte hard disks) were connected to a room-sized 24 bit DDP-224 computer programmable by its users in FORTRAN IV and DAP 24 bit assembly language. Also accessible (as subroutines residing in Fortran IV libraries) were what might be called "soft" or "virtual" input devices (random number generators, attack-decay interpolators, and a sophisticated periodic function generator) and output devices (storage buffers, including arrays for logical switches and data of different types)*. (Spiegel, 1998a)

Mathews saw the function of the GROOVE system as being a compositional tool which the composer / conductor manipulates in real time: *"The composer does not play every note in a (traditional) score, instead he influences (hopefully controls) the way in which the instrumentalists play the notes. The computer performer should not attempt to define the entire sound in real time. Instead the computer should retain a score and the performer should influence the way in which the score is played... the mode of conducting consist of turning knobs and pressing keys rather than waving a stick, but this is a minor detail... The programme is basically a system for creating, storing, retrieving and editing functions of time. It allows the composition of time functions by turning knobs and pressing keys in real time: it stores the functions on the disk file, it retrieves the stored functions (the score), combines them with the input functions (the conductor) in order to generate control functions which drive the analogue synthesizer and it provides for facile editing of functions via control of the programme time..."* (http://www.obsolete.com/120_years/machines/software/)

Remarking on the uses of the GROOVE system inside Bell Labs, Spiegel says: "*By the mid-1970s, the system had been put to many highly individualized uses. Max Mathews often programmed the system for his pioneering work on realtime control parameters and devices for performance of pre-composed musical repertoire. Emmanuel Ghent's work with GROOVE ranged from synchronous control of sound and theatrical lighting, and explorations of algorithmic motives variation, to implementation of a computer-controlled analog reverb. Dick Moore wrote a fugue generator, among other algorithms. My own use the system focused on the development of realtime control variable sets and transfer function logic for realtime computer assisted composition and improvisation*". (Spiegel's webpage)

### 2.2.4 Music Mouse

*Music Mouse is not a sequencer, but rather a mouse-driven music generator* (Keyboard Magazine, Spiegel's webpage). Music Mouse is an "intelligent instrument" written by American composer Laurie Spiegel in 1986. She describe it that way:

> "*It lets you use the computer itself as a musical instrument, played by moving the mouse with one hand while you control dozens of available musical parameters from the Mac's "qwerty". It's a great musical idea generator, ear trainer, compositional tools, and improvising instrument. The software does a lot of harmony handling for you (you control the variables it uses for this), so it's useful - as are all "real instruments" at any level of musical training, experience, or skill, from beginning through professional*". (Spiegel's webpage)

Spiegel was involved in what it could be considered as the bird of the term "intelligent instrument": "*My own first memory of the term "intelligent instrument" dates from about 1973, when Max Mathews, Emmanual Ghent, Dick Moore and I used it in discussing our work. Well before realtime digital synthesis had become established (from 1977 onwards), using hybrid (computer controlled analog) technology, we were among the very first to be able to hear the results of high level music-generating software in realtime and therefore to be able to interact with software processes while they were actually computing the music we heard, instead of listening to the stored results later. Because of the realtime nature of the system we used, this Bell Labs group may well have been the first origin of the concept, the practise of software based "intelligent instruments" and also of the expression*". (Spiegel, 1987)

### 2.2.5 The Sal-Mar Construction

In 1969, along with a group of engineers and musicians at the University of Illinois, American composer Salvatore Giovanni Martirano (1927-1995) began work on the design and construction of the *SalMar Construction*, an electronic composing / performing system that Science Digest called "the world's first composing machine". The instrument is a hybrid system in which TTL logical circuits drive analog modules, such as voltage-controlled oscillators, amplifiers and filters.

The performer sits at a horizontal control panel of 291 lightable touch-sensitive switches (no moving parts). The two-state switches are used by a performer to dial sequences of numbers that are characterized by a variety of intervals and lengths. A sequence may then bypass, address, or be added to other sequences forming an interlocked tree of control and data according to a performer's choice. The unique characteristic of the switch is that it can be driven both manually and logically, which allows human / machine interaction. The most innovative feature of the human / machine interface is that it allows the user to switch from control of macro to micro parameters of the information output. This is analogous to a zoom lens on a camera.

The information output is converted from digital to analog form and is routed to oscillators, filters and amplifiers, whose output is sent to one or more of 24 speakers. Four groups of sounds with independent control of route and rate can be distributed among the 24 speakers so that a traffic of sound is created in the space. All sounds are produced in real-time as the composer / performer according to his own prerogatives chooses a route and functions through a store of pre-programmed information. Although it was a digital machine, it was not included a general purpose computer. As in the case of the *Electronium*, the *Sal-Mar Construction* was guided through an improvisation rather than played, in the traditional sense.

Interactive composing pioneer Joel Chadabe states that although some of the elements of "interactive composing" are evident in works by other composers, Salvatore Martirano's *SalMar Construction* exemplifies the primary characteristics of the approach. Martirano performs in reaction to what he hears, manipulating aspects of music, such as pitch, rhythm, tempo, pattern, octave, spatial distribution, and cycling. (Chadabe, 1984)

### 2.2.6 The CEMS System

In 1967, while director of the Electronic Music Studio at State University of New York at Albany (1965 - 1998), American composer Joel Chadabe designed the CEMS (Coordinated Electronic Music Studio) System, an analog-programmable electronic music system built by Robert Moog. Unlike the *Sal-Mar Construction*, the CEMS was designed to be used on the studio. Since this time, the concept of interactive composing has grown in his work. He describes how the concept was born while performing with the CEMS:

> *Because I was sharing control of the music with the sequencers, I was only partially controlling the music, and the music, consequently, contained surprising as well as predictable elements. The surprising elements made me react. The predictable elements made me feel that I was exerting some control. It was like conversing with a clever friend who was never boring but always responsive, I was, in effect, conversing with a musical instrument that seemed to have its own interesting personality*. (Chadabe, 1997a)

Chadabe proposed a definition of Interactive Composing as a two-stage process that consists of (Chadabe, 1984):

1- Creating an interactive composing system, and
2- simultaneously composing and performing by interacting with that system as it functions.

Creating the system involves bringing together a computer, sound generator, and at least one performance device, and programming the computer with algorithms that function automatically and in real time to:

1- Interpret a performer's actions as partial controls for the music
2- Generate controls for those aspects of the music not controlled by the performer
3- Direct the sound generator

One key concept developed around these systems is the so-called "intelligent instrument":

> *An interactive composing system operates as an intelligent instrument – intelligent in the sense that it responds to a performer in a complex, not entirely predictable way, adding information to what a performer specifies and providing cues to the performer for further actions. The performer, in other words, shares control of the music with information that is*

> *automatically generated by the computer, and the information contains unpredictable elements to which the performer reacts while performing. The computer responds to the performer and the performer reacts to the computer, and the music takes its form through that mutually influential, interactive relationship*. (Chadabe, 1997a)

Chadabe developed in the eighties, together with D. Zicarelli, a successfully comercial program for Apple Macintosh named "M", which acts like an intelligent musical instrument. Its inner core is based on Markov chains. On the first page of the user manual we can read: "*M is an interactive composing and performing system that takes notes and chords that you specify and manipulates them, under your control, to create musical compositions which unfold during live performance*". (Chadabe, 1997b)

In Chadabe's opinion, "*the CEMS System and the SalMar Construction were the first interactive composing instruments (...) These instruments introduced the concept of shared, symbiotic control of a musical process wherein the instrument's generation of ideas and the performer's musical judgment worked together to shape the overall flow of the music*" (Chadabe, 1997a).

### 2.2.7 Some early comercial "intelligent instruments"

In (Spiegel, 1987), the author makes some historical explanations on intelligent instruments. In the Winter 1986-7 issue of Computer Music Journal was stated that the release of two programs ("M" and "The Jam Factory" by Intelligent Music, Inc.) was "*historically important because it marks the first time that intelligent musical instruments are available to the general public*". Spiegel disagrees with CMJ because, in her opinion, "*IM's programs were not the first available. There are several historical precedents to the IM programs*". Besides some unspecified examples and her **Music Mouse** system, she cites several precedents:

**Instant Music:** by Robert Campbell (1986). This program acts much like a stencil superimposed on mouse movement in one dimension, producing pitches in realtime which are subsets of predetermined chord progressions or scales provided with the program as components of supplied pieces (Spiegel, 1987).

**Sequencer**: by Dave Oppenheim (1985). This program permits realtime interactive change of the transformations to which recorded materials are subjected while replayed. Several types of transformation, involving transposition, rhythmic and durational variation, and melodic permutation are changable on-the-fly in realtime. The Sequencer also allows generation of new musical sequences from others through various realtime-changable transformations (Spiegel, 1987).

**Dr. T's:** by Emile Tobenfeld, Jack Deckhard, and Jim Johnson (1984). Algorithmic and compositional programs. While intended for composition rather than performance, these are still relevant precedents. Amongst the "Dr. T." group of programs occur the abilities to change interactively in realtime such parameters as transposition, time delay, orchestration, and the random seeds used in realtime algorithmic music generation (Spiegel, 1987).

**Algorithmic Music Language:** by Ray Jurgens (1981). Though intended primarily as a medium of algorithmic definition for non-realtime (compiled) compositional use, it was modified and updated in 1982 to accept and use keyboard console input interactively during realtime play. This software has been used for realtime-modifiable algorithmic generation in live performance by Jeff Rona and other members of its user community (Spiegel, 1987).

**McLeyvier:** by David McLey et al. Functioned as an intelligent instrument via its interpretive macro language. Individual macros were triggered by specific user actions, allowing, for example, complete reorchestration during live performance in different ways depending on the specific pitches played, selective retrieval in realtime of materials from disk for playback as accompaniment depending on keyboard performance content, random pitch generation, and macro sets for music education which changed the instrument's configuration and response depending on individual students' abilities to match pitches or accomplish other musical tasks (Spiegel, 1987).

**Muse:** by Marvin Minsky (early 1970's). It was a self-contained "computer" instrument for melodic generation and thematic development (Spiegel, 1987).

**Modular Analog Synthesizer:** by Don Buchla (circa 1964). An additional very important category of precedent which is often overlooked because of its non-digital logic was the modular analog synthesizer. Though it's often assumed that "intelligent systems" must be digital, the analog computer, which dealt with relationships among informational components "by analogy" (as analogues of each other) has much in its logic and design which should not be overlooked or forgotten. The complex realtime interactive generative and transformational processes which these instruments permitted place them as the very first "intelligent instruments" ever to be made publically available (Spiegel, 1987).

## 2.3 Emerged ideas from fractals and chaotic systems

As have been demonstrated by many composers, it is possible to model on a computer different features of music in accordance with a certain mathematical function of arbitrary origin. This function could be taken from any sonic model, vibratory phenomena or any kind of model found in nature. *Earth Magnetic Field* (1970), by American composer Charles Dodge, translates to pitches the daily variations of the Earth magnetic field. The human speech controls the rhythm and modules the sound pitches in *Cascando* (1980), also by Dodge, in *Verbes pour cuellir* by Marc Battier, and in *Erosphère* (1982), by François Bayle. A very recent example is *Erwin's Playground* (Fischman, 2003) where the author applies the Schrödinger's Equation for an atomic potential with radial symmetry (a dynamic system) to asynchronous granular synthesis techniques.

One of the scientific models that has deserved most attention from musicians and researchers in the past years has been the concept of *Fractal Geometry* (Mandelbrot, 1975, 1983). Spectacular and impressive, fractal images have conquered an important space inside visual arts and fantastic films. But, how this field of modern mathematics has been involved in music? Let's see first what fractals are.

The term *Fractal Geometry* was first introduced by mathematician and IBM researcher Benoit Mandelbrot in his 1975 book *Les Objets Fractals* (Mandelbrot, 1975), and widely known with the publication of his famous *The Fractal Geometry of Nature* (Mandelbrot, 1983). This book, where are defined and presented the fundamental concepts of this new geometry, successfully popularized several aspects of fractal geometry, such as the beauty and attractive of fractal images.

The word *fractal* comes from the Latin word *fractus*, which means fraction. By creating this new word, Mandelbrot wanted to classify and name a kind of object which owns a fractional dimension. The objects from Euclidian geometry, to which we all are familiar, own an integer dimension. For instance: a point has dimension zero, a line has dimension one, an area has dimension two. Mandelbrot calls this dimension *topological*. In the case of fractal objects, their dimension is greater than their topological dimension.

On a first attempt, Mandelbrot defines the new concept: *A fractal is by definition a set for which the Hausdorff-Besicovich dimension strictly exceeds the topological dimension* (Mandelbrot, 1983). This definition requires a definition of the terms "set", "Hausdorff-Besicovitch dimension" and "topological dimension", which is always integer. On the other hand, this definition excludes sets that should be considered fractals. Mandelbrot later gives a new definition, less formal from the mathematical point of view: *A fractal is an object made of parts similar to the whole in some way*.

There have been some propositions of fractal definition. Intuitively, we can consider as a fractal, a rough or fragmented geometric shape that can be subdivided in parts, each of which is (at least approximately) a smaller copy of the whole. Fractals are generally self-similar and independent of scale; i.e., a bit looks like the whole, and no matter how close you zoom in, they look similar. These images own, almost literally, an infinity complexity; they reveal more and more details, without limits, when we zoom in. Many mathematical structures are fractals; e.g. Sierpinski triangle, Koch snowflake, Peano curve, Mandelbrot set and Lorenz attractor. Fractals also describe many real-world objects that do not have simple geometric shapes, such as clouds, mountains, turbulence, and coastlines.

Fractal images are generated by means of relatively simple calculi, repeated again and again, using recursively on each step the results of the previous step. In many cases they are calculated by means of *Iterated Functions*. The numbers generated by an iterated function exhibit three types of behavior: steady-state, periodic, and chaotic. Chaotic functions are special cases on nonlinear dynamic systems. They represent deterministic processes that are very sensitive to initial starting conditions. That is why the theory that studies such systems is often referred to as the chaos theory, which is:

> *The qualitative study of unstable aperiodic behavior in deterministic nonlinear dynamical systems.* (Kellert, 1993)

Since the publication of Mandelbrot's *The Fractal Geometry of Nature* (1983), some musicians and researchers have investigated and used the ideas behind the theory. These applications range from real and non-realtime symbolic data generation, to music structure modeling, melody variations, or sound modeling / synthesis. They are largely based on the concept of self-similarity, as well as on iterated functions.

### 2.3.1 Melody and 1/*f* fractal noise generation

Perhaps the earliest application of fractals to music composition has been to melody construction, after the suggestions appeared in (Mandelbrot, 1975, 1983). There exist three kind of so-called fractional noises, whose spectrum diminishes following the formula $1/f^y$, where f represents frequency, and $0 \leq y \leq 2$. They have been widely used as music random generator.

The first one is called *white noise* ($1/f^0$), where there is no association between a note and the next one. It is quite unpredictable. The second one is called *brown noise* ($1/f^2$). Music generated with $1/f^2$ noise moves along from one pitch to another within a small span of intervals. It seems to wander around with no clear direction, though it is too predictable. *Pink noise* ($1/f^1$) is the third one. Its behaviour fall between white and brown noise, and is generally known as 1/f noise.

1/f noise (one-over-f-noise) appears in nature all over the places in many ways: as variations in annual amounts of rainfall, in patterns of sunspot activity, as noise in electronic devices, in traffic flow, radioactive decay, chemical systems, granular flow, ecological systems, human speech and even in music.

In fact, physicists Richard F. Voss and John Clarke analyzed several recordings of music in various styles such as Bach's Brandenburg Concerti, Vivaldi's four seasons, etc, and found the loudness and frequency distribution was nearly 1/f.

> *The spectral density of fluctuations in the audio power of many musical selections (...) varies approximately as 1/f (f is the frequency) (...) This result implies that the audio-power fluctuations are correlated over all times in the same manner as "1/f noise" in electronic components. The frequency fluctuations of music also have a 1/f spectral density at frequencies down to the inverse of the length of the piece of music. (...) The observations on music suggest that 1/f noise is a good choice for stochastic composition. Compositions in which the frequency and duration of each note were determined by 1/f noise sources sounded pleasing. Those generated by white-noise sources sounded too random, while those generated by $1/f^2$ noise sounded too correlated.* (Voss, 1978.)

The music derived from the 1/f noise is the most closed to the human music: it does not have the unpredictability and randomness of white noise nor the predictability of brown noise. 1/f processes correlate logarithmically with the past. Thus the averaged activity of the last ten events has as much influence on the current value as the last hundred events, and the last thousand. Thus they have a relatively long-term memory.

1/f noise is a fractal one; it exhibits self-similarity, one property of the fractal objects. In a self-similar sequence, the pattern of the small details matches the pattern of the larger forms, but on a different scale. In this case, is used to say that 1/f fractional noise exhibits statistical self-similarity. The pink noise algorithm for generating pitches was first described by Martin Gardner (Gardner, 1978) and has become a standard in algorithmic music. Despite the general acceptance of the results given by Voss and

Clark, the methodology and general conclusions have been under critic but constructive exam. In (Nettheim, 1992) the author concludes:

> *The claim of Voss and Clarke that 1/f processes well represent pitch in music has been found in these preliminary studies of classical music to have only slender support, and the claim for duration must evidently be rejected. Some apparent confusion involving the separation of melodies into pitch sequences and duration sequences has been pointed out, and it is suggested that melody is more appropriately analysed as their single-sequence resultant, particularly if spectra are to be calculated. In the present studies of melodies so defined, the spectrum has been found to tend more towards the 1/f-squared than the 1/f function, for periods up to about four bars of music. (...) Although these conclusions are on the whole negative, it is hoped that they may clear the way for work on other characterizations having a stronger musical basis.* (Nettheim, 1992)

Nevertheless, it seems that 1/f algorithm for note generation will remain well-regarded among musicians, because it has been widely tested and compared with the other two mentioned fractional noises: "*listeners found the music derived from 1/f processes preferable, the white processes producing music considered too random and the 1/f-squared too highly correlated*" (Nettheim, 1992).

### 2.3.2 Music structure and self-similarity

In 1983 American composer Charles Dodge writes *Profile* (Dodge, 1988), an electroacoustic work elaborated from an interpretation of the self-similarity concept in the way it is present in one of the most famous fractal objects: the Koch curve (a.k.a. Koch snowflake). Conceived by mathematician Helge Von Koch in 1904, this curve looks like a star of many peaks, where each side is infinitely composed of the same element. This curve is built through an iterative process:

Let's begin with a simple equilateral triangle. Now, on each of its three edges, equilateral triangles are added whose edges length 1/3 of the original edge. At this moment the figure has 6 peaks and 12 edges. If we do the process again, that is, on each of the 12 edges we add triangles whose edges length 1/9 of the original, we will obtain a figure of 48 edges. If we now add triangles whose edges length 1/27 of the original the same way we did previously, the resulting figure will has 192 edges. We could infinitely apply this procedure. If on each step we calculate the border length, we will note that on each generation this length will increase in 1/3. So, infinitely applying the process, again and again, we will obtain that the border length tends to infinite, while its area will tend to a finite value. On infinite, the border dimension falls between one and two, so it is a fractal one.

Dodge conceived a musical structure based on a metaphorical interpretation of the self-similarity concept. He departed from the Koch curve for building parallel voices that contain proportional relationships between them, similar to those existing between its triangles (Dodge, 1986, 1988). Note the difference between a graphic, which exists in a two dimensions space (plane), and music, which elapses in time. That is why in the Dodge's algorithm is established an analogy between the triangles that conform the

Koch snowflake, and the musical notes. Long notes are related to big triangles, and short notes are related to little triangles. He used the 1/f algorithm for creating a three voices electroacoustic piece, where each line exhibits statistical self-similarity. Musical details, such as pitch and timing, are generated with this algorithm. *Profile* is recursively time-filling in the same way the Koch curve is recursively space-filling.

> *Charles Dodge's composition for tape alone, "Profile", is an algorithmic composition in which the choice of all the elements of pitch, timing, and amplitude were made by the systematic application of 1/f fractional noise. Dodge thinks of the work as a "musical fractal" in that the structure of the work exhibits multiple levels of scale and self-similarity.* (Dodge, 1997)

For a detailed description of the algorithm, see (Dodge, 1988).

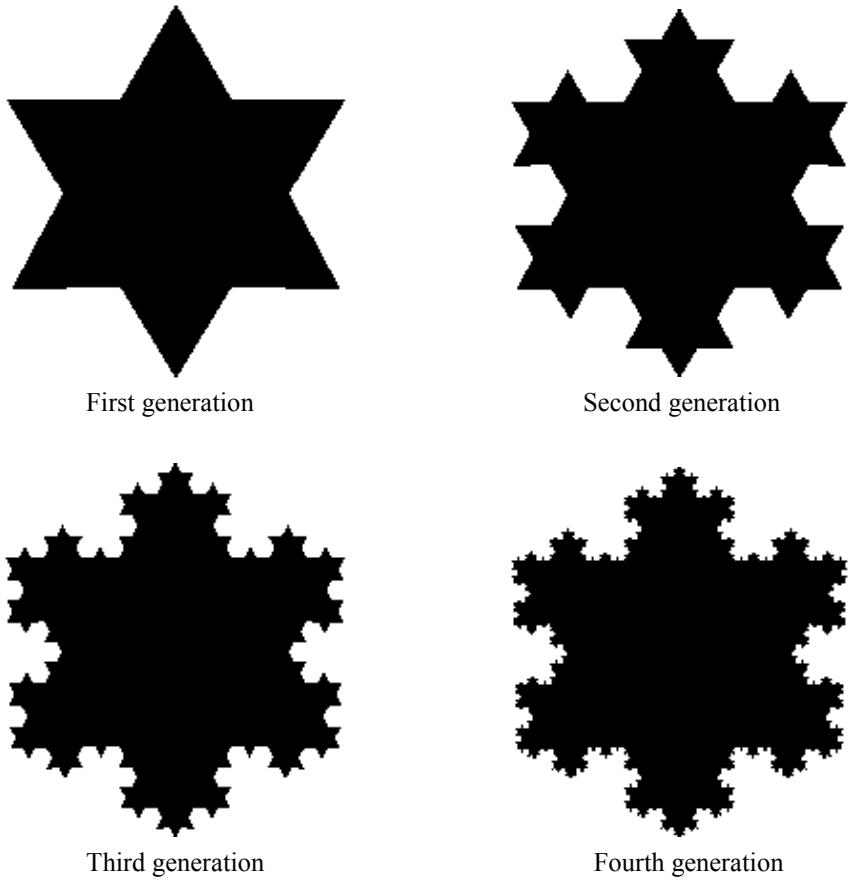



**Figure 2.1** The first four generations of the Koch snowflake.

### 2.3.3 Another applications

Another field that has been influenced by fractal theory is sound synthesis. This application is beyond the scope of our research. For further information, the reader could refer to the related bibliography such as (Chapman, 1996), (Monro, 1993), (Polotti, 2001), (Truax, 1990), (Waschka, 1989) and (Yadegari, 1991).

Another interesting application is the performance of musical variations from a chaotic mapping. MIT researcher Diana S. Dabby proposed a way of doing so. She explains:

> *A chaotic mapping provides a technique for generating musical variations of an original work. This technique, based on the sensitivity of chaotic trajectories to initial conditions, produces changes in the pitch sequence of a piece. A sequence of musical pitches $\{p_i\}$, i.e., any piece ranging from Bach (or earlier) to contemporary music, is paired with the x-components $\{x_i\}$ of a Lorenz chaotic trajectory. Each $p_i$ is marked on the x axis at the point designated by its $x_i$. In this way, the x axis becomes a pitch axis configured according to the notes of the original composition. Then, a second chaotic trajectory, whose initial condition differs from the first, is launched. Its x-components trigger pitches on the pitch axis (via the mapping) that vary in sequence from the original work, thus creating a variation. There are virtually an unlimited number of variations possible, many appealing to expert and nonexpert alike.* (Dabby, 1996)

Another possible application is the realtime generation of musical materials by means of iterated and chaotic functions. As it was aforementioned, these functions are simple dynamic systems which exhibit a complex behaviour, becoming chaotics. The idea is to plot (or at least compute) the functions and simultaneously generate control data to drive a sound synthesizer, or to control parameters of a realtime synthesis algorithm. I will tackle this approach (my approach) in next chapters. For more information about fractal and chaotic systems application to music composition, the reader could refer to papers such as (Bidlack, 1992), (Bolognesi, 1983), (Degazio, 1993), (Di Scipio, 1990), (Fagarazzi, 1988), (Gogins, 1991) and (Nagashima, 1993).

## 2.4 Review of related literature

I found a high and variable amount of publications on the following related areas: Algorithmic Music Systems, Interactive Music Systems, Mapping, Human Computer Interfaces, and Fractal Music. I also found a few papers relevant to Intelligent Musical Instruments. Perhaps the nearest papers are Joel Chadabe's *Interactive Composing: An Overview* (CMJ, 1984), and *The Multimedia Instrument* (2001), a three pages article where the author describes the kind of instrument I call "active". In fact, Chadabe states in this paper that "*whereas the link in an acoustic instrument is essentially **passive**, offering no more information than that which is specified by the performer, the link in an electronic musical instrument can be **active**. The link, in other words, can itself generate musical information that complements or shares control of the musical process with the performer*" (bold is our).

Regarding the use of chaotic functions as generative algorithms of musical materials, I found about ten scientific papers. Two of them (a P.h.D. thesis and an article from the same author) face the problem of generating musical variations from existing pieces. Among the whole papers, the oldest dates from 1988, and the newest is from 1995. I also found two P.h.D. dissertations which seem relevant to this research work. To my mind, I should undoubtedly read these three dissertations. They are:

1- Bidlack, R. 1990. “Music from Chaos: Nonlinear Dynamical Systems as Generators of Musical Materials”, Ph. D. dissertation, University of California, San Diego, 1990.
2- Manzolli, J. 1993. “Nonlinear Dynamics and Fractals as a Model for Sound Synthesis and Real-time Composition”, Ph. D. dissertation, University of Nottingham, 1993.
3- Dabby, D. 1995. “Musical Variations from a Chaotic Mapping”. Ph. D. dissertation, Massachusetts Institute of Technology, 1995.

## 2.5 Conclusions

In this chapter, several hardware and software examples of early projects that exhibit the main notions of Active Instruments, have been mentioned and reviewed. A common idea is the existence of a self-regulated sonic system, which owns a personal sonic behaviour that can be controlled in realtime in an interactive way. Obviously, interactive realtime algorithmic music systems do exist since many years ago. Thus, any system that fit the proposed model could be called active instrument, though I have not mentioned any recent development. It has been also presented some possible applications of fractal and chaos theories to music composition, because I have chosen iterated and chaotic functions as a particular and narrow approach to the study of active instruments. In Chapter 1 the reader will find six differents reasons for doing so. Finally, a brief review of related literature was exposed.

# Chapter 3

# My previous related work

*Los músicos siempre, de alguna forma, hemos utilizado algoritmos; no en el sentido este, moderno, pero nosotros siempre hemos utilizado pasos, ordenamientos, lógica... para hacer una obra musical. Incluso, hemos utilizado cualquier tipo de pensamiento matemático para ver los problemas de la estructura de una obra, para construir determinada armonía.*

Roberto Valera
(Hinojosa, 1997)

## 3.1 Introduction

As it was aforementioned, in the first stage we made several experiments *toward the general study and investigation of algorithmic music*. We started from scratch, almost without foreign references and some difficulties for accessing technical and scientific information. Additionally there were no previous experiences in our national context. Nevertheless, we were influenced by the work of some people. We took from Charles Dodge our very first approach to algorithmic music; from Max Mathews the idea of interactivity; and from the Barrons, Bischoff and Perkis we reinforced the notion of a self-regulated sonic system, which owns a personal sonic behaviour, that can be controled in realtime in an interactive way. The carried out experiments lead us to the concept of active instrument.

## 3.2 Objectives and Methodology

Our objective was *the general study and investigation of algorithmic music, aimed at the development of generative models for assistance in music composition (electroacoustic and instrumental music).* This is a very wide field, where many approaches has been experimented. At the time fractal theories were very popular, thus we focused on the investigation of possible applications of these theories to computer music composition. This way we narrowed the nature of the problem.

Work carried out consisted in the development of four projects, each one scheduled in the following four main steps:

1- Development of related theoretical principles and algorithms for the generation of music,
2- their implementation as a compositional software,

3- and their realization as one or more compositions that demonstrate their musical validity and viability, and the usefulness of the software.
4- An additional step was the analysis of previous ones, in order to get conclusions and propose future directions.

## 3.3 Experimental design

I followed a bottom-up approach, going from simple and knew things (musical scales, melodic variations…) to complex and unexperimented ones, taking into account a holistic point of view. The carried out experiments consisted in the development of the following four projects:

1- A **non-realtime** (non-interactive by nature) algorithmic music system (*Musical Fractals*). A metaphorical interpretation of the fractal's self-similar concept was experimented. I also experimented a 1/f algorithm for the generation of symbolic data (pitches). The use of traditional musical knowledge, in the form of musical scales and melodic variations, was also tested.
2- A **realtime interactive** algorithmic music system (*Orbis Musicae*). Some possibilities of realtime control were investigated on a very simple (but useful) approach. No fractal algorithm was tested. We noticed for the first time the notion of a self-regulated sonic system, which owns a personal sonic behaviour, that can be controlled in realtime in an interactive way. This notion is central to the idea of an active instrument, and as we learned later, emerged in the fifties of the past century.
3- A **realtime non-interactive** algorithmic music system (*Piano Fractal*). I experimented here the realtime generation of musical material from the computation of an iterated chaotic function. I tested only one function and a simple mapping strategy. The system was intended to autonomously generating the notes of a complete piece, thus the source code became a sort of a score. There were no interaction with the user (neither initial data), only "play" and "stop" commands were conceived. This is another example of a self-regulated sonic system which owns a personal sonic behaviour, but in this case the lack of interactive control leaves the system out of the notion of active instrument.
4- A **realtime interactive** algorithmic music system (*Fractal Composer*). This is an obvious evolution from the precedent experiments. I integrated here the experiences learned from the previous projects in the form of musical scales, simple music knowledge and interactive realtime control, plus new features such as multiple chaotic models and complex mapping strategies. Here we find a tangible example of active instrument.

On each experiment we followed the four-step methodology aforementioned. We went across four approaches to algorithmic music where time and interactivity were studied. When I refer to time and interactivity, I think about the generation process of musical material itself. These approaches were:

1- non-realtime process (non interactive by nature),
2- realtime interactive process (simple approach),
3- realtime non-interactive process, and
4- realtime interactive process (complex approach).

## 3.4 Detailed description of each project

In 1989 Cuban composer Carlos Fariñas (1934-2002) founded, with some colleagues, the *Estudio de Música Electroacústica por Computadora* (Studio for Electroacoustic and Computer Music) of the *Facultad de Música*, at the *Instituto Superior de Arte* (University of Arts) in Havana. One year later I started collaborating with the Studio as an undergraduate student of Computer Science. From the beginning I was assigned a project on Algorithmic Music. Our knowledge on the field was almost null, so we started from scratch, provided only with a couple of articles (Dodge, 1986, 1988) and some anecdotal references on the Xenakis' and Hiller's works.

### 3.4.1 Musical Fractals (1990-1994)

Fractal images were very popular at that moment; the musical experience based on fractals carried out by American composer Charles Dodge (Dodge, 1986, 1988) was a starting point for our research. Dodge suggested a musical structure based on a metaphorical interpretation of the self-similarity concept. He departed from the Koch curve for building parallel voices that contain proportional relationships between them, similar to those existing between the triangles of the Koch curve (see Chapter 2).

We elaborated and implemented an algorithm based on the Dodge's interpretation of self-similarity. Our first system, *Musical Fractals*, which runs under MS-DOS, needs as a seed data a melody, a list of melody transformations, and some numerical values such as: number of voices, values that will affect the relationships between them, etc. It computes the "piece" in non-real time, generating up to four parallel voices. One of the voices is the original melody and its variations. We also implemented some interesting features that proved, through practical experiences, its strength and weaknesses. Some of these features are:

1- **Scales**. The program is able to use up to fifteen different musical scales, even a user defined one, for computing the whole "piece".

2- **Traditional melodic variations.** Melodic transformations from classical counterpoint, such as: *inversion*, *retrograde*, *augmentation* and *diminution*, are algorithmic procedures used along centuries of musical tradition. They are powerful tools for evolving a melody, so we decided to test their potential inside a computer program.

3- **Non-traditional melodic variations.** Some non-traditional melodic variations were implemented, following very personal approaches. They are:

   **Addition:** Randomly adds some notes to the melody without affecting the total length.

   **Subtraction:** Randomly deletes some notes from the melody without affecting the total length.

**Reverse time:** It is like traditional retrograde, but it only reverses the note durations of the melody.

**Reverse pitch:** It is like traditional retrograde, but it only reverses the note pitches of the melody.

**Generation:** Uses a 1/f fractal noise generator for replacing each pitch of the melody.

**Simulation:** Uses a particular approach, based on *Markov Chains,* for replacing each pitch of the melody. The resulting melody sounds a little bit like the original one.

**Arpeggio:** Replaces the notes whose duration is greater than or equal to a quarter-note, by an arpeggio of four notes, without affecting the total length of the melody. The algorithm uses interval values provided by the user.

**Logarithmic:** Replaces every pitch by a new one, computed with a personal algorithm that uses the logarithmic function, and involves all existing pitches.

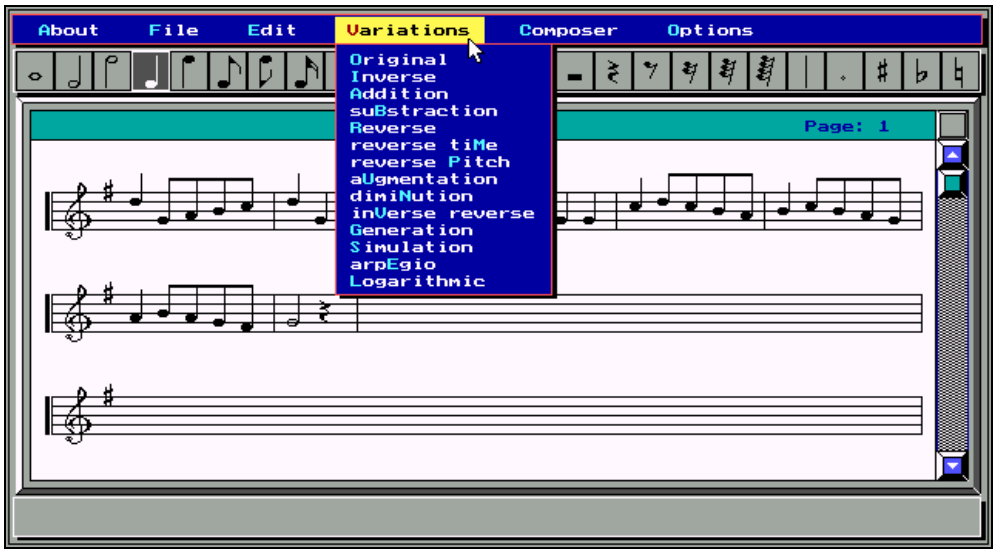



**Figure 3.1** Screenshot of Musical Fractals.

An interesting feature that proved good results is the possibility of applying not only isolated melodic variations to the melody, but a set of joined variations that conform a much complex transformation. For instance, think about applying the following variations in order: augmentation, arpeggio, simulation and diminution. The resulting melody is only one, not four. Of course, it is possible to obtain four different melodies too! A simple command interpreter was implemented for entering coding of complex transformations. Composers found this possibility as a new and powerful compositional tool. A particular effort was the creation of an embedded score editor (by Claudio Daniel Ash) for entering the melody and for another upcoming projects. The resulting "piece", as well as the original melody and every transformation can be heard through a

Roland MPU401 MIDI interface and an external sound module. Additionally, they can be saved to a Jim Miller's *Personal Composer* MIDI file (Miller, 1985).

## Some Details of the Implementation

*Musical Fractals* was implemented in Borland's Turbo Pascal, and uses the features that environment has just offered for object oriented programming and modularity. Among others, the system includes functions for:

- **Processing transformations:** The user should write down in a window a list of letters, which represents compounded transformation commands. For instance, the following variations: augmentation, arpeggio, simulation and diminution, would become the string "uesn". There is a function which interprets the list of command strings when the composition process starts running.


**Figure 3.2** Screenshot of Musical Fractals' transformations interface.

- **Executing the composition algorithm:** The composition algorithm was implemented by means of a recursive function, which dynamically builds a tree that holds all the data of the resulting music but the note-lengths. This function calls a simple command interpreter which applies the transformations to the original melody. Later, another recursive routine will traverse the tree for assigning the note durations. We need to store the tree as an ordered succession of Note-On and Note-Off MIDI events, so the tree is converted into an array, which lets us apply any known sorting algorithm. Then, the tree data is saved to a temporal file, the occupied memory is freed, and the saved data is loaded again, this time into a dynamic array. Later the nodes are sorted with the ShellSort algorithm. A previous implementation used the Quicksort, but because the huge amount of data, the Stack Overflow error usually raised. At the time, memory was a very scarce resource on PCs.

- **Performing the composition through MIDI:** This function performs the final resulting "piece" through a Roland MPU 401 MIDI interface. A previous version generated the MIDI information into a *Personal Composer* Lisp program (Miller, 1985). Then it was necessary to run that amazing system and load the generated Lisp program for listening to the music, and eventually for obtaining the traditional music score.

- **Saving the "piece" into a MIDI file:** There are two functions that save the resulting "piece" into a MIDI file. The first one saves the data into a structured file designed by this author. This file could be loaded by another programs developed by us, like the score editor written by Claudio Daniel Ash, a programmer who worked at the EMEC/ISA. The second one saves the "piece" into a *Personal Composer* MIDI file. One of the faced and solved problems was the deciphering of the MIDI file structure used by the *Personal Composer*, for lack of technical information about that. This kind of file could hold score graphical information, MIDI information, or both included. In our case, the generated file only holds MIDI information.

- **Implementing a modified Markov chain:** One of the melody variation methods used by *Musical Fractals* is called Simulation, because the resulting melody sounds a little bit like the original one. It is based on a first-order Markov chain directly built in a directed graph, so there is no transition table. All possible transitions have the same probability. In this kind of graph, an edge goes from one node (or vertex), the source, to another, the target, and hence makes connection in only one direction. The algorithm has two steps: (1) melody codification (or training) and (2) melody simulation. In the first step the directed graph is built, where sequences of all two contiguous pitches from the original melody are represented. Initially a graph node is created for each pitch. Then, one or more edges go from each node to some other, representing this way different series of pitch pairs. In other words: we have a graph node with an associated pitch value $x$, and a set of nodes $Y = \{y_1,...,y_n\}$ also with associated pitch values. This representation let us generate, in the simulation step, several pitch pairs of the form $(x, y_i)$, where $y_i \in Y$. There are also represented some pitch pairs of the form $(y_i, x)$. These pairs do exist in the original melody; that is why when the new melody is generated with the same interval relationship from the original melody, they will sound us quite similar.

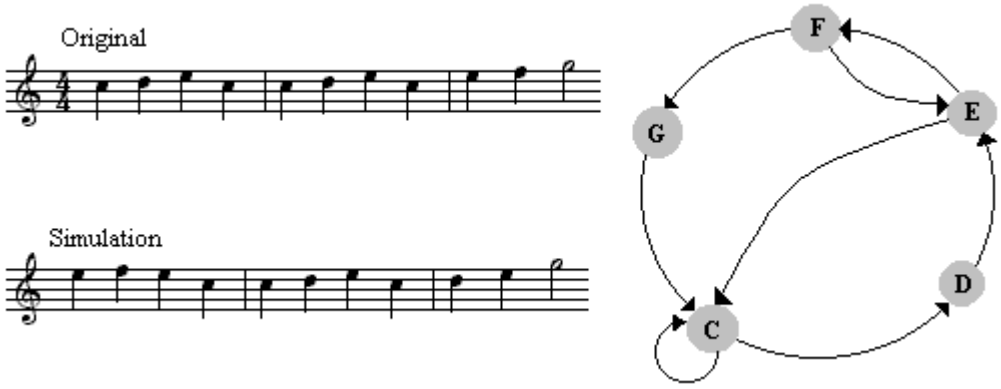



**Figure 3.3** Schematic representation of the simulation graph.

For the simulation step we depart from the first note in the original melody. Starting from any node we traverse the graph, replacing each pitch in the original melody by the pitches from the graph, just to arrive to the original melody last but one note. When

traversing the graph, it is possible that several edges go from the present node. In this case we select randomly one edge, so each simulation generates similar but different melodies.

- **Handling musical scales:** *Musical Fractals* is able to use up to fifteen different musical scales, even a user defined one, for computing the whole "piece". In the case of the User Scale, the system offers a configuration window for reconfigure the scale. They are internally represented by an array of boolean values, and are the following:

| | |
|---|---|
| Major | 101011010101 |
| Chromatic | 111111111111 |
| Minor (Harmonic) | 101101011001 |
| Doric | 101101010110 |
| Phrygian | 110101011010 |
| Lydian | 101010110101 |
| Mixolydian | 101011010110 |
| Aeolian | 101101011010 |
| Pentatonic C | 101010010100 |
| Pentatonic G | 101001010100 |
| Pentatonic D | 101001010010 |
| Pentatonic A | 100101010010 |
| Pentatonic E | 100101001010 |
| Whole Tones | 101010101010 |
| User Scale | 111111111111 |

**Table 3.1** Musical Fractals' musical scales.

*Musical Fractals* runs under MS-DOS and never was ported to MS Windows. This project provided us with our first experiences in experimenting computer algorithms for music composition. Important ideas that will affect future projects were developed, such as the use of musical scales. One electroacoustic music piece was composed by Carlos Fariñas (*Cuarzo: Variaciones Fractales, 1991*), which was premiered in 1991 in Havana, in the frame of the Festival of Contemporary Music, and has been played abroad. I also wrote a short electronic piece entitled *ET llamando a casa (1994)*, which demonstrates some of the features of the software.

### 3.4.2 Orbis Musicae (1993-1996)

In 1993 we started another project with new goals in mind. We were faced with the problem of creating an interactive music system for realtime performances. The idea came about through several ways, but a very influencing one was our personal meeting with Dr. Max Mathews in 1991, during the International Festival of Electroacoustic Music held in Varadero beach. There I had the opportunity to talk with him. Among several questions, I posed this one: *How would you use a music made algorithmically?* Mr. Mathews kindly answered:

> *I would be interested in keeping an interaction with the algorithm; a part from the computer and a part from myself. I am interested in algorithms for improvising. With these algorithms, the musician and the computer play the music together. The algorithm chooses the notes, but the musician can select, among the options given by the program, the one he likes*. (Hinojosa, 2003)

Eventually two years later and with these ideas in mind, we gave birth to our next project: *Orbis Musicae*. This is an interactive realtime algorithmic music system which acts like an instrument, where the musician controls different and variable parameters while the music is computed live. It puts a step forward in our research work, and is supported on our previous experiments. For instance, it includes the musical scales (philosophy and source code) developed for *Musical Fractals*. The basic algorithm is quite straightforward:

There are twelve planets around the Sun moving each one on an elliptic trajectory. At the beginning each planet is assigned a grade from the chromatic scale. Then one or more triangles are placed over the orbital plane. When the planetary system starts moving, one or more planets visit the area of the triangles. As soon as a planet goes in or out from a triangle, a Note On or Note Off message is triggered, sounding on or off a MIDI controlled external sound source. The next time this planet goes inside the triangle, the original pitch assigned to it could remain the same, or could be changed to a new one, according to the initial choice of the musician. Each triangle has assigned a particular MIDI channel, so different MIDI programs (instruments) or sound modules can be controlled at the same time. The musician can change the position and speed of the planets during the realtime performance. The configuration of the system, say: planet positions, planet speeds, assigned pitches, triangles and its assigned MIDI channels, can be saved internally at any time, and can be restored also whenever the user wants. *Orbis Musicae* uses ten memory banks, and follows the "total recall digital mixers" philosophy. The composer can use this feature for creating a scheme of configurations, which would be useful for planning the development of his piece in a sort of a score.

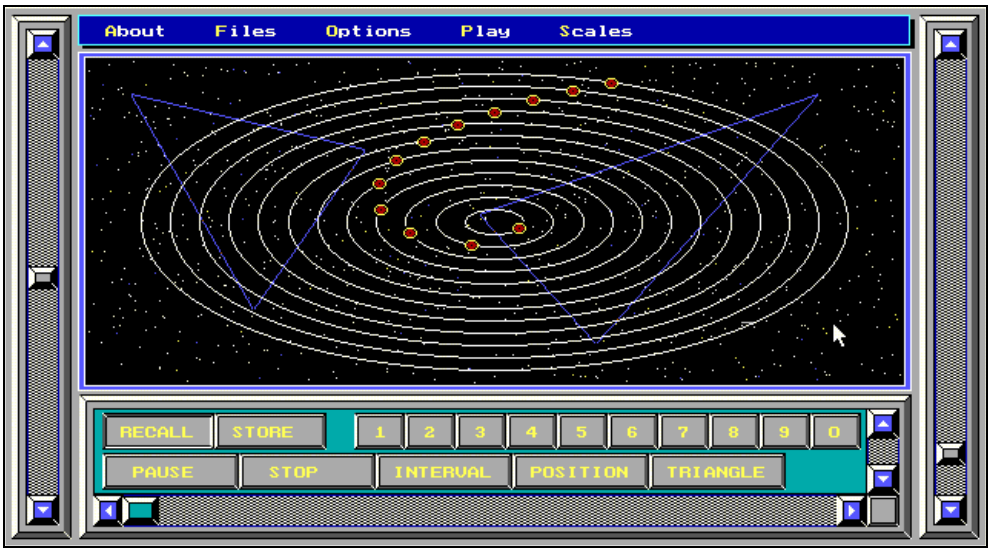



**Figure 3.4** Screenshot of Orbis Musicae.

*Orbis Musicae* has two essential properties:

1- It is a self-regulated system that has a personal behaviour. It can play itself endlessly without any human intervention.

2- The task of the human player is to influence the behaviour of the system, as if he were an instrument player. In fact, he is an instrument player. A player of a new kind of instrument, an active instrument. Traditional instruments always play a passive role, they react to the human gesture, but they are unable to offer the musician any musical idea by itself. At that time we used to name this kind of software active instrument, "virtual instrument".

In the middle of our investigation, we found previous experiences from other researchers whose works connect deeply to, and reinforce, the ideas we were working on. These experiences come, in one hand, from Louis and Bebe Barron (see Chapter 2), and on the other hand, from the work of American composers John Bischoff and Tim Perkis. At the time (1996), they became our theoretical references.

On the CD *Artificial Horizon*, recorded between 1989 and 1990 by John Bischoff and Tim Perkis, is exposed a sample of what they call "Music for New Software Instruments". In the CD booklet they express the philosophy of their music in the following terms:

> *For us, composing a piece of music is building a new instrument, an instrument whose behavior makes up the performance. We act at once as performer, composer and instrument builder, in some ways working more like sculptors than traditional musicians. (...) There is another feature of the computer that attracts us: its ability to build systems of interaction with complex dynamics, systems only partially predictable, which can develop a unique "body" of their own. These woolly computer instruments can also be designed to respond to players' actions in new ways, creating a music which contains the trace of human gesture, in addition to having a degree of autonomy. In fact, for us, the distinction between composing a new piece of music and building a new instrument is not clear-cut: composing a piece of music for us IS building a new instrument, an instrument whose behavior makes up the performance. We act at once as performer, composer and instrument builder, in some ways working more like sculptors than traditional musicians. And in each case, the focus is on creating a system as open and alive as possible, bearing the precious marks of an individual character.* (Bischoff, 1990)

And specifically talking about his 1978-80 piece *Audio Wave*, John Bischoff says:

> *AUDIO WAVE was written for pianist Rae Imamura (...). My idea was to make a live computer piece for Rae where both of her hands would be continually active, as in her conventional keyboard playing, but where her actions would serve to influence an ongoing musical output rather than have the task of initiating each sound* (Bischoff, 1991).

When we knew about the principles behind the works by the Barrons, Bischoff and Perkis, and after looking back to our experiences, we felt that we had found what path to travel through. So, we decided to build:

> An interactive algorithmic music system for realtime performances, not-based on any known musical style, which could act as an active instrument (self-regulated system), *where the user's actions would serve to influence an ongoing musical output rather than have the task of initiating each sound*.

*Orbis Musicae* runs under MS-DOS and, as well as its predecessor, never was ported to MS Windows; furthermore, never was finished a non-prototype version. It constitutes our first approach to realtime interactive algorithmic music systems, or what we simply prefer to call Active Instrument.

An electroacoustic music piece was composed by Carlos Fariñas, who used its realtime capabilities for recording fragments of music in a sequencer. Later he took these fragments for creating a tape composition (*Orbitas Elípticas, 1993*). This music work was played in 1994 in the Bourges' International Festival of Electroacoustic Music. Another electroacoustic music work was created by Cuban composer Roberto Valera, who used the software for his realtime piece *Hic et Nunc* (Eli, 1998), performed for the first time in 1996, with my assistance, in the frame of the Havana's Festival of Contemporary Music

### 3.4.3 Piano Fractal (1996)

After experimenting with the previous programmes, we decided to start a new research approach, and look back again to fractals. We wanted to design a new system where the previous developed ideas were presented, and much closer to the mathematics of chaos. So, we got closer to fractal images and to the process of generating musical material live from the realtime computation of an iterated chaotic function, although in non-interactive way. I implemented the computation of an iterated function and the generation of MIDI control data (i.e., note-on and note-off) simultaneously with the visual plotting of this function. I started experimenting the Henon mapping, which can be drawn by iterating the following functions:

$$Xn+1 = Xn * Cos(Alpha) - (Yn - Xn^2) * Sin(Alpha)$$
$$Yn+1 = Yn * Sin(Alpha) + (Yn - Xn^2) * Cos(Alpha)$$

I tested only one function and a simple mapping strategy. There were no interaction with the user (neither initial data), only "play" and "stop" commands were conceived. This is another example of a self-regulated sonic system which owns a personal sonic behaviour, but in this case the lack of interactive control leaves the system out of the notion of active instrument. The first experiments were so encouraging that, after some simple improvements, I made my first "serious" electroacoustic music piece, named *Piano Fractal* (1996). The computer controlled live an Akai S1000 sampler with piano sounds, while I applied sound effects on a Yamaha DMP7 digital mixer. The used source code could be considered as the score of the piece.

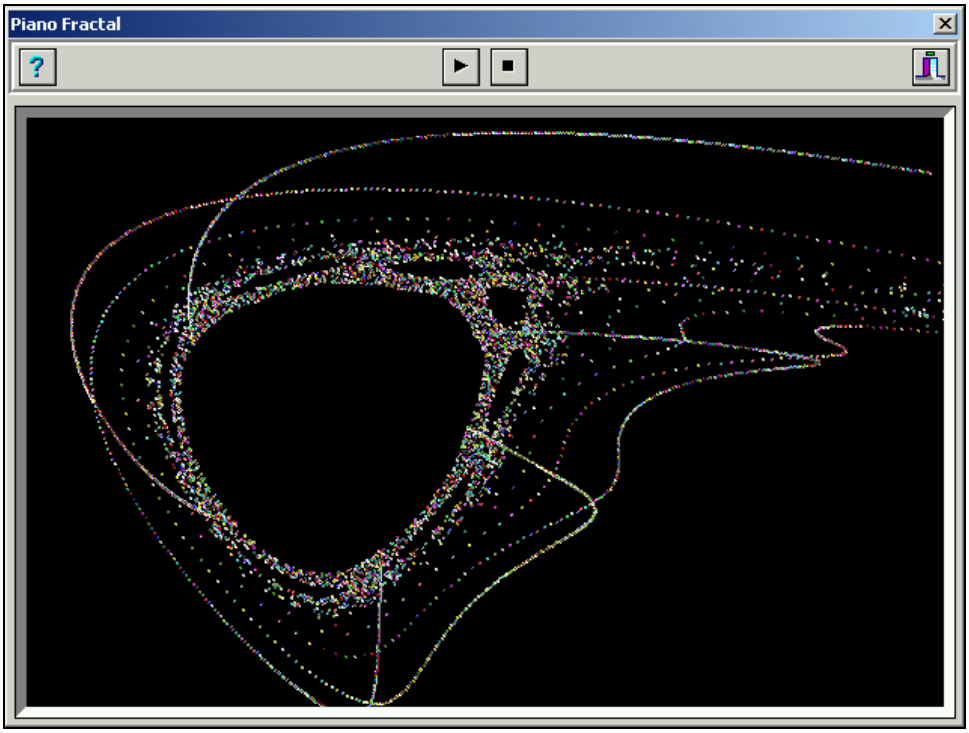



**Figure 3.5** Screenshot of *Piano Fractal*, a software that generates the notes of my piece of the same title.

The interface had no interactive control, so the question came up: What if I could modify the behavior of the system in realtime? That was the genesis of Fractal Composer.

### 3.4.4 Fractal Composer (1996-2000)

*Fractal Composer* is intended to be a virtual musical instrument for realtime performances. It plots chaotic attractors, dynamic systems and some related formulas, and makes music from these calculations while the musician introduces changes to musical parameters and listens to the results, all of this in realtime.

The system, which runs under MS Windows, features seven different fractal formulas and related algorithms for tone generation, which combine six ways of mapping pixel colors into pitches, and four note-duration or rhythms. Up to four interdependent voices may be used, conducted through three different manners or styles. Each voice owns its loudness or dynamics, its pitch limits (range) and scale. This program offers twenty-four different scales, including nine user defined ones.

The user may have control over some MIDI functions like: program changes, modulation and panning. Each voice can be moved from left to right or vice versa, automatically, at the speed the user chooses. Although program names are showed in the General MIDI convention, of course it is possible to use any non-GM external sound module. In addition to this, it is possible to load a digital sound file and play it together with the MIDI fractal music.

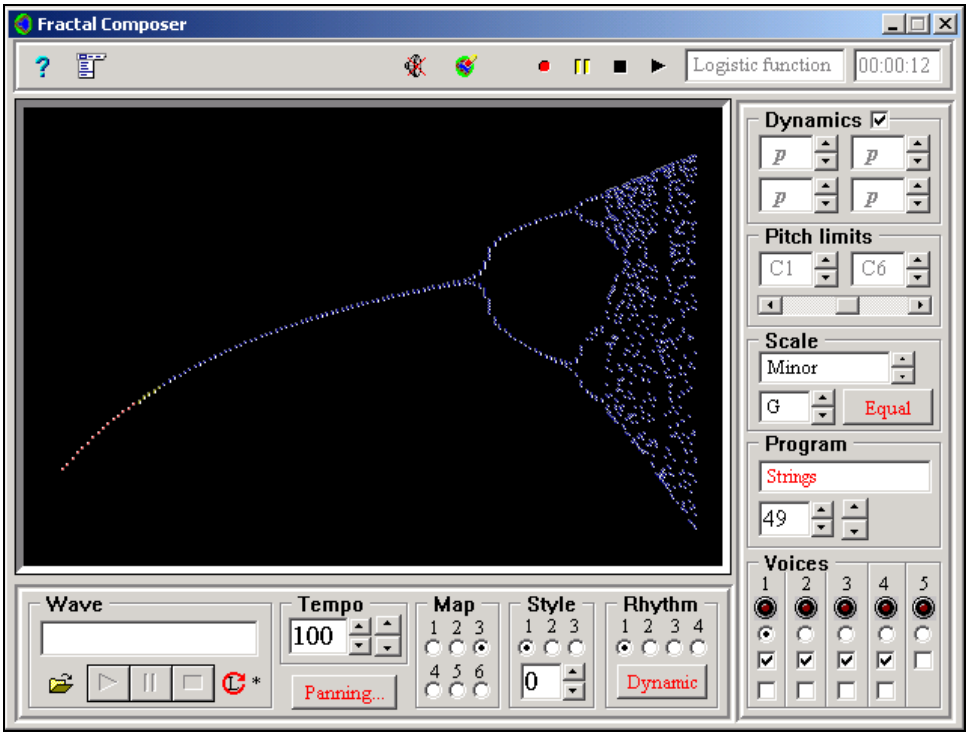



**Figure 3.6** Screenshot of Fractal Composer.

While *Fractal Composer* creates music in realtime, it is possible to save the resulting music, with all the changes (performance) the user has done, in a Standard MIDI File. This lets him edit his music in any sequencer or music notation software that supports SMF. The musician can store all the settings in a configuration file to be recalled later, in another session. This means that the player doesn't lose his fractal type nor its parameters, voices selected, patches, dynamics, scales, and even his own scales. A chronometer appears in the upper right-hand corner of the display to inform the performer about the duration of his piece as time goes by.

As Xenakis said in 1971: *With the aid of electronic computers, the composer becomes a sort of pilot: pressing buttons, introducing coordinates, and supervising the controls of a cosmic vessel sailing in the space of sound, across sonic constellations and galaxies that could formerly be glimpsed only in a distant dream.* (Xenakis, 1992)

## Some details of the user interface

*Fractal Composer* is a mouse-driven virtual instrument. It can be guided by interacting with the controls the user graphical interface shows. These controls allow to influence the ongoing musical output in a playing or recording session. Some of them are:

- **Dynamics:** Controls the loudness of each voice.
- **Pitch limits:** Defines the pitch limits (*tessitura*) of each voice.

- **Scale:** The user can choose one of the twenty-four different scales, including nine user-defined ones. If the right button reads *Equal*, the same scale is applied to all the voices. If that button reads *Not equal*, each voice is owner of its own scale.

- **Program:** Allows to choose an instrument for each voice. Program names are showed in the General MIDI convention, though any non-GM external sound module can be controlled.

- **Panning:** When the *Panning* button is pushed, a dialog box becomes visible showing four sections. Each panning section is coupled to one voice. They can be moved from left to right, from right to left, or from left to right to left, automatically, at the speed the user chooses.

- **Mapping:** The mapping section lets the user to choose one of the six preset formulae to translate pixel colors into pitches. It is influenced by the selected style, or the way the voices are combined, which is actually another step inside the implemented mapping.

- **Rhythm:** *Fractal Composer* uses four different note durations, which can be statics or dynamics. Inside each voice the durations could be the same for a while, or they could be changed automatically. Here the voices keep proportional relationships similar to the used in *Musical Fractals*, based on the Charles Dodge's interpretation on self-similarity.

## Used functions

All the formulae share two common parameters: *R factor* and *Buffer*. The buffer is a note list that may be repeated in accordance with the *R factor* value. The R factor influences how many times the buffer is repeated.

- **Henon mapping:**

  Xn+1 = Xn * Cos(Alpha) - (Yn - Xn^2) * Sin(Alpha)
  Yn+1 = Yn * Sin(Alpha) + (Yn - Xn^2) * Cos(Alpha)

  This is a deterministic algorithm; i.e., with the same initial parameters, always sounds the same notes.

- **Henon Attractor:**

  Xn+1 = 1 + Yn - a*Xn^2
  Yn+1 = b*Xn
  Ex.: a = 1.4 b = 0.3

  This is the strange attractor most often linked with Michel Henon, an astronomer at Nice observatory in southern France who came to the subject of fractals while studying the dynamics of stars moving within galaxies. The Henon attractor is an

example of a very simple dynamic system that exhibits strange behavior. This is also a deterministic algorithm.

- **Logistic function:**

  $Xn+1 = a * Xn * (1-Xn)$
  $0 < a \leq 4$

  This is a classic chaotic generator. It is a deterministic algorithm. The Logistic function (a.k.a. Logistic map or Feigenbaum map) is one of the most studied non-linear equations in chaotic dynamics. It models very well the behavior of many natural phenomena such as: predator-prey ecosystems, food-population, etc.

- **Julia set:**

  $Z = Z^2 + C$
  Fix C and move Z across the complex plane.
  Ex.: C = -0.74543 + i0.11301, $-1.65 \leq real(Z) \leq 1.65$, $-1.65 \leq imag(Z) \leq 1.65$, iterating $Z = Z^2 + C$ for each Z and test it for convergence. Ex.: $| Z | \leq 2$.

  This set was named after mathematician Gaston Julia, and can be generated by a simple change in the iteration process. As C is a complex number, there is a Julia set corresponding to every point on the complex plane, thus there is an infinite number of Julia sets. It is possible to zoom-in on the Julia set image. This is a deterministic algorithm.

  - **1/f noise (Sometimes called Pink noise):** The following three formulas are knew as fractional noises, whose spectrum diminishes following the formula $1/f^y$, where f represents frequency, and $0 \leq y \leq 2$. They are stochastics. As they involve random processes, each time are played the notes will be different but the sonic result will be similar. The 1/f algorithm used by *Fractal Composer* is adapted from the one in (Roads, 1996 p. 884).

  - **White noise ($1/f^0$ noise):** This is the first and most basic random process. There is not any relationship between a value and the next one, and all of them have the same probability to be generated. This is not a fractal formula, but it is related with 1/f noise.

  - **Brown noise ($1/f^2$ noise):** The brownian music moves along from one pitch to another within a small span of intervals. It seems to wander around with no clear direction. This is not a fractal formula, but it is related with 1/f noise.

*Fractal Composer* integrates several principles (and even source code) developed on its two precedent projects. For instance, it takes from *Musical Fractals* the use of musical scales, the 1/f function, and the rhythm proportions between the generated voices, inspired on the self-similarity concept and conceived by Charles Dodge. It takes from *Orbis Musicae* the concept of realtime control for interactive performance / guiding of a self-regulated system.

There have been several approaches to algorithmic music based on iterated functions. One of them could be the plotting of a dynamic system while MIDI control data, i.e. note-on and note-off, are generated. Starting from this basic idea we distinguish three branches:

1- Design of a graphical user interface where *mathematical* (non-musical) parameters from the inside equations are represented, as a way of control.

2- Design of a graphical user interface where *musical* parameters are represented as a way of control, which hides the mathematical core.

3- Design of a graphical user interface where both previous points are developed in a balanced and reasonable way.

In our approach we have chosen the second point, because in one hand, there is only one interface for controlling (musically) several different mathematical models, where each of them owns a different set of parameters. On the other hand, this system is designed to be used in the easiest and intuitive way. For instance, it is not necessary to have any knowledge on fractal or chaos theory to use it on a first lower level. The musician can start from scratch just by interacting with the traditional musical concepts reflected on the interface, such as: *dynamics*, *pitch limits*, *scales*, *voices*, *tempo*, etc. In fact, this capability is pointed out in the user manual, where the author hints the user:

> *To start, simply push the Play button and listen to the music for a while. After that, let's do some changes. Push any button or control you want and hear the results immediately. This is a good starting point. For the future I recommend you to make a composition plan intended to be performed in public, or to be recorded and saved into a standard MIDI file for later edition in a sequencer or in a score edition software for creating, finally, your own composition. Another starting point is to listen and examine the example configuration files. Keep in mind that you can listen to these files and create SMFs without introducing changes, nevertheless the aim of the system is to be guided.* (Hinojosa, 1999)

*Fractal Composer* has been presented in Cuba, in public demonstrations as well as in concerts. It has also been presented to a scientific meeting there (Hinojosa, 2000; Vidal, 2000). Spanish composer Andres Lewin Richter has just used the software for writing a piece for tape and piano (August, 2003). The author wrote three electroacoustic music pieces with this system:

1- *El fin del caos llega quietamente*, which is intended to demonstrate that mathematics can also be a path to music, and also aimed at demonstrating partial progresses of our experiments. It was created entirely in real time from the calculus of the *Logistic Function*, and recorded in one pass with no overdubbing.

2- *Satélites*. Basic sonic material was created in real time from the calculus of the *Henon Map*. It was premiered in the XII Havana's Festival of Contemporary Music, in 1997.

3- *Oro-iña*, a realtime performance for computer, Afrocuban percussion and two dancers. It was played for the first time in 1998, in the frame of the International Festival of Electroacoustic Music "Spring in Havana" (Vidal, 2000).

**Figure 3.7** Picture of the Oro-Iña performance (Photo: Archie).

## 3.5 Conclusions

In this chapter we have summarized the first stage of our research work on algorithmic music. Actually the investigation went through several steps, in which many experiments and searches were carried out, many practice problems were solved, an in-depth study on music theory was done, and some experiences were built-up while simultaneously new algorithms for handling the essential elements of music were conceived and tested.

Derived from many experiment, non-artistic music fragments were generated, which lead us toward the improvement of our systems. At the same time, while satisfactory partial results were obtained, they were applied to the composition of artistic works of music by well-regarded composers who trusted, from the beginning, in the creative capabilities of primary versions for doing their compositional labour. These pieces have demonstrated the musical validity and viability of the tested approaches, and the usefulness of each software.

Each experiment taught us some aspects of the nature of the problem, ranging from aspects of algorithmic music through to interactive control. Eventually the notion of active instrument emerged. I found partial answers to the aforementioned questions that arise from the analysis of the general structure of an active instrument. Although I was limited in the access to international scientific information, the present period of work gives me much better conditions for comparing and contrasting my research with another ones, and to propose future directions for deepen in the nature of the problem.

### 3.5.1 Example of present application

An application example of our experiences was the design and implementation, in 2002, of a graphical user interface for a project carried out inside the MTG (Robledo, 2002). At the request from Chilean-Spanish composer Gabriel Brncic, and based on his proposed algorithm, we developed a realtime audio processing system to be used live on a concert in Radio France, Paris (Brncic, 2002).

While Enrique Robledo wrote the processing core as an application of a framework developed in the MTG (CLAM, http://www.iua.upf.es/mtg/clam/), I was assigned the task of designing the graphical user interface. In *Ronde Bosse*, just like in my previous *Orbis Musicae*, it is built-up the "total recall digital mixers" philosophy as a way of scheduling realtime interactive wide changes on the configuration system. My 1993 MS-DOS program uses ten memory banks, while the 2002 Linux program incorporates up to one hundred memory banks.

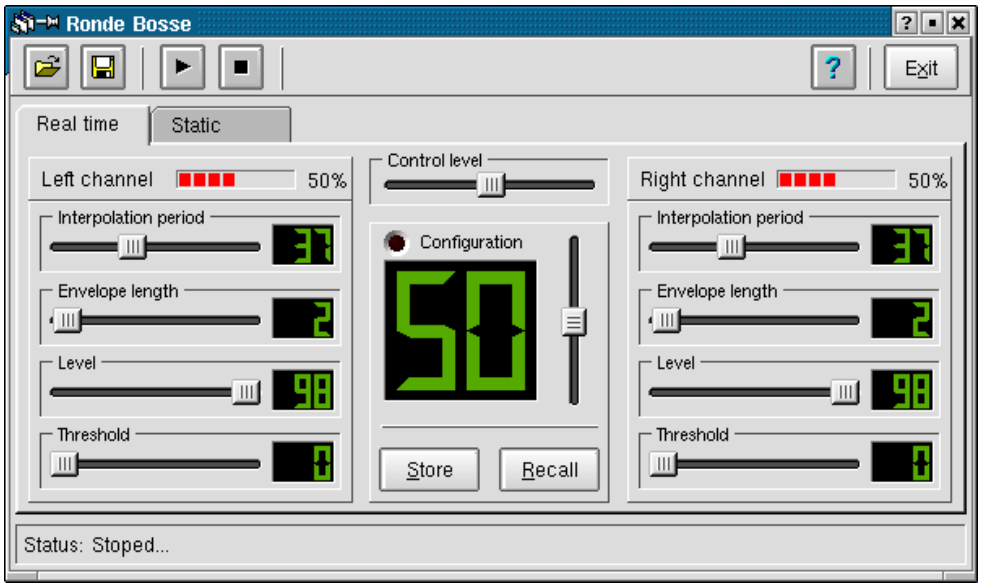



**Figure 3.8** Screenshot of *Ronde Bosse*.

# Chapter 4

# Discussion

*La machine d'arithmétique fait des effets qui approchent plus de la pensée que tout ce que font les animaux; mais elle ne fait rien qui puisse faire dire qu'elle a de la volonté, comme les animaux.*

Blaise Pascal
Pensées (N° 627)

## 4.1 Introduction

The central argument of this research work is that since the fifties of the past century has emerged the idea of a self-regulated sonic system that owns a personal sonic behaviour. This kind of system automatically proposes the composer musical materials live, in realtime, while she can influence this sonic behaviour in an interactive way. In Chapter 2 several hardware and software examples have been mentioned and reviewed. From the observation of these examples plus the analysis of my own related work, I found common notions that establish a relationship between all these projects. From my point of view, the existence of these common notions suggests that today there exists an underlying model of computer-based instrument that needs to be studied.

I intend to contribute to the development of a formalized framework within which instrument that fit the proposed model may be discussed, compared, contrasted and evaluated. In (Chadabe, 1984, 2001), (Bischoff, 1990, 1991), (Rolnick, 1992), (Rosenboom, 1992), (Rowe, 1993), (Schloss, 1993), and (Spiegel, 1987, 1992, 1998a) it is possible to find some discussions around the main notions that build-up the proposed model, sometimes under the denomination of Intelligent Instruments or even Multimedia Instrument (Chadabe, 2001). There are a lot of approaches for the generation of symbolic music data (Roads, 1989; Papadopoulo, 1999). I have chosen the generation of interactive realtime symbolic music data resulting from the application of chaotic systems, as a specific and narrow approach to the problem of study the proposed model. Chapter 1 (1.4) exposes six reasons for this choice.

As it was aforementioned (Chapter 1), the objective of this research work is *to contribute to the field of interactive realtime algorithmic music systems by identifying and addressing* a number of several questions that arise from the analysis of the proposed structure of Active Instrument, building upon the hypothesis that new meaningful sonic results and instrument-performer relationships can be achieved. Througout this chapter I intend to give partial answers to some of these questions as a first attempt in achieving, to a limited extend, the objective.

## 4.2 What is Algorithmic Music?

The high grade of abstraction and formalism that music has on its theoretical aspects, has often lead to be compared to mathematics. Not without reason, one of the most important music disciplines, Harmony, is used to be raised to the category of a science. It has been stated, for instance, that musicians have invented the notion of coordinates before mathematicians did it. In fact, a music score is actually a coordinate system, where frequencies values are represented onto the ordinate axis, and the time flow is represented onto the axis of abscissas. A musical score could be see as a mathematical structure suitable to be read, and therefore:

1- to be played again and again; that is, to convert it into a physical and temporal-bounded object;
2- to guarantee its existence through time;
3- to apply algorithmic / mathematical transformations on it.

In his creative labour, the music composer works with the so-called elements of music, such as: melody, harmony, timbre, articulation, dynamics, form, etc. These elements actually conform a very big set from which musicians extract ordered subsets structured musically. These subsets are commonly known as scores.

In Music Composition, just like in Computer Science, an algorithm is a detailed and finite sequence of instructions that, in a finite time, carry out a certain task in a satisfactory way. In this case the algorithm should generate musical notes (finite set) or numerical values (also a finite set) for electronic sound synthesis. A simple example is the creation of a melodic line by means of an algorithm that selects, according to a certain criterion, note-pitches belonging to a particular musical scale. The instructions of such an algorithm (generate-and-test) could be written as follows:

1- Work out in detail a set of rules that define a criterion for selecting note-pitches.
2- Randomly select one pitch from the C major scale.
3- Take the rules and apply it to the candidate pitch obtained in step 2.
4- If the pitch is rejected, go to step 2.
5- Add the pitch to a list of values, which will conform the desired melody.
6- If more pitches are needed, go to step 2; otherwise end.

For the note-lengths we could use a similar algorithm, or we could take equal length for every note. Note that I have left out, in search of simplicity, some elements such as articulation, dynamics or timbre. The generation of numerical values suitable to be interpreted as musical parameters, can be achieved by means of several ways. Virtually any mathematical formula can be converter into a generator of musical values.

Cuban composer Carlos Fariñas (1934-2002) used to say that every melody but those from monodic systems, always has an implicitly harmonic context. So, according to this idea, every random procedure for creating melodies (including the aforementioned) should take this principle into account. It has no sense to look for an algorithm for creating “beautiful” or “inspired” melodies without influencing the random process by a harmonic progression.

The described procedure for music composition is often referred to as *Algorithmic Composition* because a computer, according to a certain algorithm, creates a sound object musically structured. It should be understood as a sound object, the sound representation on a traditional music score, or on MIDI files, or on transmitted MIDI data, or the physical generation of that sound in realtime or its storing into a digital sound file.

Though it is possible to manually compose music by means of algorithms, without the computer intervention (recall the Mozart dice game or the early works by Xenakis), the term *Algorithmic Composition* is unavoidably used to be linked to the usage of a computer. I would like to propose the following definitions:

> **Algorithmic Music**: is the music created by a computer system in an autonomous or semiautonomous way.
>
> **Algorithmic Composition**: is a music composition procedure for creating algorithmic music.

Implicit in these definitions is the use of an algorithm for music composition, because a computer system (software) is the implementation of an algorithm in some programming language (normally is the implementation of many algorithms or sub-algorithms).

From a mathematical point of view, music composition could be defined as the process of selecting, from a finite set of elements (pitches, lengths or rhythms, timbres, dynamics, etc.) a subset, also finite. The elements of this subset should be combined and ordered according to a preconceived formal logic.

## 4.3 What is an Active Musical Instrument?

Although algorithms can be used for calculating and sending data in realtime from the computer system to a synthesizer, the procedures of classic algorithmic composition generally imply the repeated process of generation and later revision and editing of the resulting score. This way separates the acts of composing and performing, and is often referred to as nonreal time. From the moment the musician introduces the initial data to the moment the computer works and gives the results, there exists a time long enough to not consider this a realtime process. But, what if the process of composing and performing were in realtime? What if we could interact live with this realtime process?

During the past century many new musical instruments were invented where a common denominator is present; they are (or were in many cases) electronic instruments. Let us recall, for instance, the Theremin, the Hammond organ, the Moog synthesizers or the Yamaha DX7. These new instruments added novel sound colors and new sonic possibilities, but its performing, if we compare it with traditional instruments, did not introduce a new situation: the player acts over the instrument, and this one generates the indicated sound, accurately and in realtime.

Computer music systems are an important element for conceiving a new class of musical instrument, provided of some functions which have been regarded in the recent past as "intelligents". This new instrument could be able to collaborate with the musician in the music creation or performing, or even in the experimentation and searching of new aesthetic music concepts. This new kind of instrument could be named *Active Instrument.* When I talk about an active instrument, I think of a means of music realization that cooperates live, in some way, with the musician in the generation, processing and organization of sounds. The term *active* actually means that the instrument participates in an active way in the music composition / performing process, in contrast with the *passive* role that a musical instrument assumes during its playing in the traditional way. As it was discussed in Chapter 2, the notions of such a new instrument have been developed since the fifties of the past century. Obviously, this "new" instrument is not that new. What it would be new is the formalized model I intend to contribute. Let me propose the following tentative definition:

An **Active Musical Instrument** is a computer-based instrument for realtime performances / composition, who interacts actively with the musician. The system automatically proposes musical material in realtime, while the user's actions would serve to influence this ongoing musical output rather than have the task of initiating each sound. The instrument acts like a self-regulated system with a personal sonic behaviour, and its core is actually a realtime algorithmic music system.

The word "active" comes as opposite to the passive role of traditional instruments, they react to the human gesture, but they are unable to offer the musician any musical idea by itself. A characteristic feature is the lack of pre-recorded or stored sequences, they are generated in realtime. The inherent cause-and-effect aspect of traditional musical performance could become less clear. When playing an active instrument, the performer becomes a sort of orchestra conductor. The system generates musical material, but the user influences the way this material is generated. His influence acts over aspects such as: when, how and even where. There is a response to the performer's actions in terms of sonic behaviour modification. Even no user influence might be considered as a kind of influence.

Figure 1.1 represents the proposed structure of an Active Musical Instrument. From the examination of this diagram it is clear that a realtime sound synthesis / processing system, that proposes the user ongoing sonic patterns, and is able to be "guided" and influenced interactively, could be classified as an active instrument. Obviously, the model may accept a wide range of approaches, and leaves opened many doors to walk through. I have chosen the generation of symbolic music data as a particular and narrow approach. Thus, throughout this essay I adopt this specific point of view.

## 4.4 Active Instruments and Interactive Systems classification

In his book *Interactive Music Systems* (Rowe, 1993), Robert Rowe provides a framework within which interactive systems may be discussed and evaluated. There he proposes a rough classification system for interactive music systems, "*not simply to attach labels to programs but to recognize similarities between them and to be able to identify the relations between new systems and their predecessors*".

This classification system is built on a combination of three dimensions. The first one distinguishes *score-driven* systems from those that are *performance-driven*, which do not have a stored representation of the music. Clearly Active Instruments are closer to the later. The second dimension groups response methods as being *transformative*, *generative*, or *sequenced*. From the beginning of this essay, the generative nature of Active Instruments is remarked. A final third distinction is established between the *instrument* and *player* paradigms. It will be clear that Active Instruments should be placed in the player paradigm:

1- *Instrument paradigm systems are concerned with constructing an extended musical instrument: performance gestures from a human player are analyzed by the computer and guide an elaborated output exceeding normal instrumental response. Imagining such a system being played by a single performer, the musical result would be thought of as a solo.* (Rowe, 1993)
2- *Systems following a player paradigm try to construct an artificial player, a musical presence with a personality and behavior of its own, though it may vary in the degree to which it follows the lead of a human partner. A player paradigm system played by a single human would produce an output more like a duet.* (Rowe, 1993)

## 4.5 Active Instruments vs. Intelligent Instruments

The term "Intelligent Instrument" seems to be born in the seventies inside Bell Labs, when Mathews, Spiegel and other colleagues worked on realtime music systems, specifically the GROOVE program. In Laurie Spiegel's words, the system was ideal "*for the development of what we called "intelligent instruments". (...) It also made the system ideal for the exploration of compositional algorithms*" (Spiegel, 1998a). Such a term, like another ones, comes after the necessity of communication between we human, as she explains:

> *With regard to the idea of a "meaningful taxonomy" of interactive computer-based methods of musical creation, it should be borne in mind that terms such as "intelligent instrument", "algorithmic composition", and "interactive composition program" came into being at a point in computer music history when the primary reason for such terminology was the simple need to communicate to others what unprecedented things we were trying to do, in the absence of any established terminology describing creative automation in music.*
>
> *Differences implied by such terms were often highly influenced by now long-past technological limits such as realtime computer throughput: An "instrument" was something with could generate sonic response interactively in realtime. "Algorithmic composition" often resulted in printed note-lists or non-realtime sound computation, precluding interactivity. "Intelligent instruments" were distinguished from other computer-based performance instruments by their greater use of encoded decision-making logic relative to stored pre-specified note/event data.* (Spiegel, 1992)

She also proposes that intelligent instruments are "*essentially musical instruments for which the ratio of the amount of information the music system generates to the amount the musician plays, per unit of time, is greater than one to one*" (Spiegel, 1998a). Obviously, this feature also apply to my model of active instrument. In fact, I suggest that the term Active Instrument is best suitable for such systems, although not all intelligent instruments could fit the proposed model.

At the time computers were esoteric artefacts, they seemed to own some human intelligent properties (in fact, they have), but these primary elements of human thinking, such as memory and the ability to handle information, were often exaggerated. Computers were even referred to as "electronic brains", despite the enormous distance between the human brain and present computers (not to talk about earlier computers). If we examine descriptions of so-called intelligent instrument, we will realize that they mostly do not refer to what we actually call "intelligence", though it is possible to find some references to artificial intelligence techniques like expert systems / rule based systems. That is why I regard the term "intelligent instruments" somewhat inappropriate. Although quite long to be completely cited, I would like to bring here the following very interesting paragraph from (Rolnick, 1992):

> *The primary question I have related to the second theme mentioned above is what points or transitions are relevant to musicians along the continuum between completely passive instruments and autonomous (algorithmic or knowledge-based) performing machines? What do the various buzz words used in this area (e.g., "hyperinstruments," "intelligent instruments," "interactive composition systems") really mean, if-as I doubt-they mean anything at all (outside the marketing departments of companies or the public relations departments of institutes of technology)? Where does one draw the line between an instrument with a stored procedural response to pre-specified input (my understanding of "hyperinstruments"), and instruments which generate material based on user interaction with a model of their compositional algorithms (my understanding of "interactive composition systems")? Are there meaningful definitions of "composition" and "performance" whereby the new instruments and performance situations make sense (and the term "interactive composition" is made an oxymoron)? How should we classify a system such as Robert Rowe's "Cypher," in which the computer software actually models a "listener" component that responds to the characteristics (at some level) of the live performer's playing? If we had a meaningful taxonomy of these systems, how could it help us to develop more flexible human-instrument interfaces and more interesting computer-based musical instruments?* (Rolnick, 1992)

From my point of view, an Intelligent Instrument should be actually "intelligent". Should be an instrument that indeed exhibits an intelligent behaviour. Such an instrument should be able to carry out logical inference (deduction, induction), to improve with experience by modifying its memory contents (machine learning), or to extract useful information (general rules, interesting patterns) from its inputs (data mining). I have found a similar point of view in some research works such as the article *Instruments That Learn* (Wessel, 1991), the paper *Agent-based Implementation on Intelligent Instruments* (Dapoigny et al., 2003), or in the CNMAT Studio Report (Andrews, 1997), where we can read:

> *III. Intelligent Instruments: neural networks and gestural interfaces. Synthesis control involves the mapping of input gestures to the parameters that control the details of sound synthesis and compositional algorithms. These mappings are most often non-linear and, as in the case of additive synthesis, usually involve a large number of synthesis parameters that must be generated from those few parameters extracted from the gestures. Neural networks provide a mechanism for this "few to many" parameter mapping problem and are being actively explored for purposes of real-time control.* (Andrews, 1997)

It follows that an Active Instrument could also be an Intelligent one, and vice versa.

## 4.6 A few words about mapping

There have been several approaches to algorithmic music based on iterated and chaotic functions (Bidlack, 1990, 1992; Dabby, 1995, 1996; Degazio, 1993; Di Scipio, 1990; Gogins, 1991; Little, 1991; Nagashima et al, 1993). One of them could be the plotting of a dynamic system while MIDI control data, i.e. note-on and note-off, are generated. Starting from this basic idea I distinguish three branches:

1- Design of a graphical user interface where *mathematical* (non-musical) parameters from the inside equations are represented, as a way of control.

2- Design of a graphical user interface where *musical* parameters are represented as a way of control, which hides the mathematical core.

3- Design of a graphical user interface where both previous points are developed in a balanced and reasonable way.

Two common elements of these branches are GUI and mathematical core. The so-called *mapping* internally links them. Mapping establishes a correspondence between the parameters involved in the mathematical core (perhaps Ifs) and musical attributes (out-direction). It also establishes a correspondence between the HCI and inner-model parameters (in-direction). Thus, mapping becomes a critical element of the whole system (see Figure 1.1).

Figure 4.2 (a simplification of Figure 1.1) depicts the function of mapping in an interactive realtime algorithmic music system based on a mathematical model. The core generates simple numbers, which are interpreted and converted by the mapping into musical values. These values are translated into music, and could be represented in the GUI. Finally, the user listens to the music and, perhaps, sees some kind of representation on the GUI, that also represents controls by mean of which he can influence the mapping configuration. This description belongs to the second branch. In the first and third approach, the user could also influence the behaviour of the mathematical core directly from the GUI.

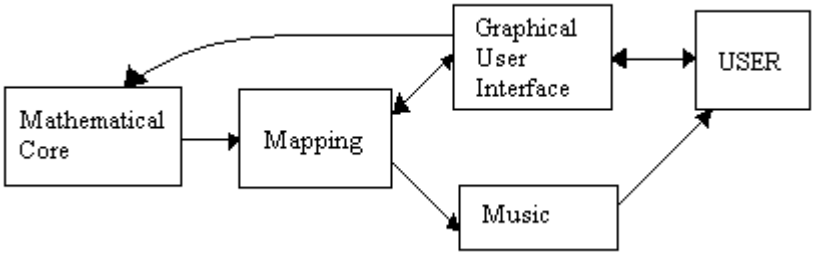



**Figure 4.2** Mapping simplified scheme.

In my approach (*Fractal Composer*) I have chosen the second branch not only in search of intuitiveness. Because the musician should handle several different mathematical models, each one with particular parameters, the realtime GUI only represents mapping parameters, which are common to every chosen generative core. The user only faces the core, in non-realtime, to configure its initial values.

Mapping is a very empirical subject. A good mapping should intend to conciliate two different areas of human activity, science and art, in a satisfactory way. It is the translator from the mathematical domain into the musical domain so, it is a very critical element of a realtime algorithmic music system based on a non-musical formalism. Because its intrinsic empiricism, it is very difficult to find a good mapping. Brazilian composer Eduardo Reck Miranda remarks on this problem:

> *Finding and effective method for mapping the orbits onto musical parameters is not an easy task. This is one of the greatest difficulties that composers face when working with algorithmic composition systems that use the output from essentially non-musical processes; (...) devising mappings that are too simplistic may strip a potentially rich orbit of its details, producing music that is dull and uninteresting. Conversely, a method that is too complex may mask the behaviour of the orbit and jeopardize the composer's original intention to use the iterative process in the first place. Clearly, a balance must be struck. (Miranda, 2001)*

## 4.7 Intuitive vs. Non-Intuitive Software

In (Hinojosa, 2001), Spanish composer Eduard Resina assesses software for computer aided composition:

> *In general, they are not very intuitive at all for traditional musicians, who basically come with knowledge or learning of many years in musical terms, in musical concepts, and software quite often does not reflects that. It would be essential to develop software where you can really work with musical concepts. Software has to be more intuitive for musicians, and certain solutions have to be found in this direction.*

From my point of view and experience, Resina's reflection is critical for the goodness of music composition software. Computer musicians often cross the boundaries of their

knowledge area, and fall into our area of sciences, so we often forget who actually they are, i.e., musicians.

They are usually pushed into dealing with lots of scientific or technical terms, so a non-computer-trained musician will get lost inside such a software environment. Perhaps we have the right to say: "This kind of musician should study computer sciences in order to use our software". To our mind, this is not a fair statement.

In my opinion, modern musical education should include some standard technical computer knowledge. At the same time, we software developers should try to keep the user interface inside this standard knowledge. I suggest there should be a reasonable balance between "standard knowledge" and "non-standard knowledge" in music software user interfaces.

When a user interface shows musical terms or concepts, it becomes more intuitive and easy to use by a trained musician. That is why I opted to carry on these ideas in my previous projects. For instance, it is not necessary to have any knowledge on fractal or chaos theory to use my *Fractal Composer* system on a first lower level. The user can start from scratch just by interacting with the traditional musical concepts reflected on the interface, such as: *dynamics*, *pitch limits*, *scales*, *voices*, *tempo*, etc. He will also be faced to some "standard knowledge" such as the standard symbols for *Record*, *Pause*, *Stop* and *Play* from tape recorders.

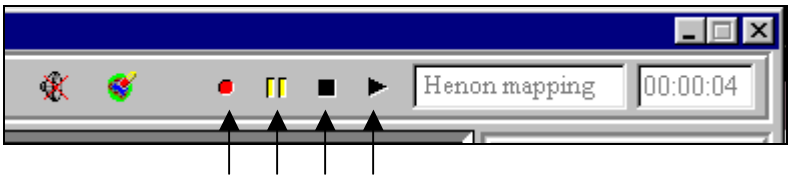



**Figure 4.3** Standard knowledge: *Record*, *Pause*, *Stop* and *Play*.

I consider the algorithm behind an algorithmic system as a way of creating music. So, what it is important to me is what musicians can do with the system, musically speaking, and not the used algorithms, to some extend. If the algorithms are so good, efficient and sophisticated, from the technical point of view, but it offers bad results from the musical or artistic point of view, the software becomes useless.

## 4.8 In search of a satisfactory algorithm

Algorithmic Composition researchers have tried different approaches for handling the music elements and for generating musical structures. Traditionally one of the most important elements of music has been melody. Many algorithms and models for "composing" melodies have been developed, from 1/f fractal noise (see Chapter 2) to rule based or constraint programming (see appendix 1). An interesting discussion on this subject can be found in (Pachet, 2000).

Cuban composer Carlos Fariñas (1934-2002) used to say that every melody but those from monodic systems, always has an implicitly harmonic context. So, according to this idea, every random procedure for creating melodies should take this principle into

account. It has no sense to look for an algorithm for creating "beautiful" or "inspired" melodies without influencing the random process by a harmonic progression.

On the other hand, when researchers intend to mimic a known music style, it is often known what elements, characteristics or procedures they should model, but what path should be follow in order to generate satisfactory musical structures not-based on any known musical style? Musicology and music composition tradition have the answer. When we were looking for a solution, composer Fernando Rodríguez (Archie) came up to us and replied: "you should try to model Analogy and Contrast".

> *The notions of foreground and background (...) are critical in controlling musical flow. If similarity is in the foreground, the listener will perceive the music as continuing uninterrupted; if difference is more prominent, then the perception will be one of contrast. (...) When contrast is in the foreground, it is introduced to avoid boredom, and to deepen the listener's experience. Contrast creates emotional breadth, setting off ideas and heightening relief and definition of character. (...) Musically, when we hear familiar material in new contexts, its meaning is enriched.* (Belkin)

The idea of analogy and contrast on music structure is also applied to evolve melody, in the form of repetition and variation. This conception, developed throughout centuries of musical practice, has been explained from the point of view of Information theory. Composers such as Lejaren Hiller (Hiller, 1981) or Laurie Spiegel (Spiegel, 1998b) have consciously used that theory on their own compositional work.

These reflections around melody and structure only refer to our western music tradition. It could be possible that they do not match with music traditions from other different cultures ( Hinojosa, 2003).

## 4.9 In search of a definitive composition system

Throughout the music history, composers have developed techniques, most of them algorithmic procedures, to handle all the elements of music, say: melody, harmony, rhythm, timbre, articulation, form… What a composition system does is to apply these techniques, and even new or personal ones, to the material provided by the user, i.e., the composer. The ways to handle the elements of music are so many, almost infinite so, from our point of view, it is impossible to find a definitive computer composition system (Hinojosa, 2001).

Every music algorithm leaves its fingerprinting in the sonic result of its execution. It has no sense to look for an universal algorithm for composing any known music style. Composers use many algorithms or algorithmic procedures everyday, and the doors for creating new compositional procedures and new music styles are always opened, though it's no easy to travel it through. Music composition involves creativity, which is impossible to lock in a scientific model. It always flies away beyond our imagination (Hinojosa, 2003).

## 4.10 Two reflections about authorship

**First**. In (Jacob, 1996) I found an interesting question that made me think: "*(...) if an algorithm faithfully represents an artist's creative process, what is the difference between music produced by the artist and music produced by the algorithm?*"

Algorithmic composition leads to the following situation: the user gives instructions to a computer to conceive an object (music). After a while, he receives this object from the machine. So, what now? He says: "this is my own work". Has the man stolen the object from the machine? Does this object belong to the computer?

Do not forget who has mentally conceived that object before its physical existence. Man has thought about that object, with more or less precision, before giving instructions to the machine. So, the computer has the task to give birth to the object dreamed by the man. When an artist designs a monumental sculpture, it is built by several (or even many) workers, but nobody has doubts about the authorship of the sculpture. Who is the author of the Sagrada Familia temple? Who denies it is Antonio Gaudí? Who denies the authorship of the Tour Eiffel to Gustave Eiffel?

Computers only simulate, through very strict instructions from the man, some elements of the human thinking. During the creative process, they can contribute some things to the task commanded by the man, but they can only contribute things that were thought before, things that were mentally conceived previously by the man. They cannot contribute things unconceived by the man, because they have no will nor awareness. That is the difference between music produced by the artist and music produced by the algorithm.

It is really very interesting that the previous conclusion came about many years ago. Things have not already changed despite the progresses on Artificial Intelligence research. French great thinker Blaise Pascal was aware of the lack of will on the machine, and about its philosophical implications. He wrote in his famous 1660 *Pensées*: *"The arithmetical machine produces effects which approach nearer to thought than all the actions of animals. But it does nothing which would enable us to attribute will to it, as to the animals"* (http://www.ccel.org/p/pascal/pensees/pensees07.htm).

The vision about that computers can only contribute things that were thought before and mentally conceived previously by the man, was astonishingly realized by Ada Byron:

> *The Analytical Engine has no pretensions whatever to originate any thing. It can do whatever we know how to order it to perform. It can follow analysis; but it has no power of anticipating any analytical relations or truths. Its province is to assist us in making available what we are already acquainted with. This it is calculated to effect primarily and chiefly of course, through its executive faculties; but it is likely to exert an indirect and reciprocal influence on science itself in another manner.* (Lovelace, 1842)

The man conceives and programs creation strategies, which imitate his possibilities, skills and knowledge. So, his personality will be present in the machine's results. Computers have no special artistic skills or virtuosity. They only have a representation

of the skills and knowledge from the man. They are only able to mimic those human properties.

Can a machine express its individuality, its own personality, its own subjectivity? These qualities are not properties of a computer, so they cannot be expressed. Only the man can express his individuality, personality and subjectivity, from the moment he selects and gives instruction to the machine, from the very instant he conceives a music program, or when he configures the options of the software. Machines impregnate with some logic and formal characteristics the result of its computations, but the man is who gives imagination to those calculi, the man is who transmits his human sensibility with the help of a computer, and he is who transforms in art the science that could exists in automatic creations ( Hinojosa, 2003).

**Second.** I have found also in (Jacob, 1996) the following interesting statement: "*(...) music produced by algorithmic composition is considered somehow inferior not because it was produced by an algorithm, but because it is someone else's music--it belongs to the designer of the algorithm, and not to the user of the algorithm.*"

If we accept this statement as a valid one, maybe it should be said: Wozzek does not belong to Alban Berg (the user of the twelve-tone algorithmic procedures) but to Arnold Schoenberg (the designer). Traditional non-computer algorithmic methods are really compositional procedures, which are always adapted by the composer to his mental scheme, to his personal point of view about music, and to his own experience and skills. When the user configures the options of any algorithmic composition system, and gives it the seed data, he transmits his own personality, as well as when he uses any conventional algorithmic procedure, or even a rule-based music composition formalism like traditional counterpoint. So, I firmly believe that music composed with the aid of an algorithmic composition system, belongs to the user ( Hinojosa, 2003).

## 4.11 Ideological considerations and motivations

Usually when talking about algorithmic music several questions arise: why would we like a machine to compose music? Do we want to end up the creative work of human composers? Will composers be extinguished just like dinosaurs did it? These questions remember the emerged doubts when the first computers were born: will the machine replace the man? "*It would not be forgotten that informatics is only a tool*", used to say Xenakis. "*If sometimes I use mathematical functions, or even physical theories in music, is because* ***do exists a deep relationship between music and numbers***" (Xenakis, 1986) (bold is our). This statement is very revealing because the mathematical feature of music makes it suitable for applying mathematical way of thinking, just like composers have been doing since ancient times, with the development of compositional (algorithmic) procedures. The computer is only a tool, a creation instrument.

Music analysis is one application of algorithmic composition research. In fact, this idea arose up since the early days. In his 1960 book *Les Musiques Expérimentales,* Dr. Abraham Moles remarks: *(...) the experiences on mechanical music composition reveal itself useful not only for composition, - after all composers do exist yet in modern world, - but mainly for analyzing, conceiving, and eventually modifying with full knowledge of*

*the facts, the rules that govern the assembly of sound objects, about which the conventional study of music was very discreet"* (Moles, 1960, translation is our).

By about the same early period (1962) French composer Edgard Varese gave a talk at the University of Yale, where he pointed out other important ideas: *"We might remember that there is no magic in the machine, as it is beginning to believe, and that we should no expect electronic devices to compose for us. Electronic means allow to compose good and bad music, as has been for traditional instruments. The computer is a wonderful invention that seems almost superhuman, but its limits are actually the limits of the individual who gives its material"* (Roads, 1986, translation is our).

Due to the wide range of possibilities offered by computers and other electronic music devices, which are sometimes exaggerated, it is often though erroneously that usual music knowledge is unnecessary for making music with those equipments. We think computers are a powerful tool for the musician. They will help the artist in developing his ideas, in stimulating his imagination, in speeding up some technical procedures of music composition. Computers enrich the compositional process, but they will not provide the user an unexisting talent. Nevertheless, they are able to stimulate the development of an undiscovered talent or innate musical capabilities ( Hinojosa, 2003).

A huge amount of publicity was generated when the *Illiac Suite* was first performed, a lot of it rather silly, according to Hiller's consideration (Hiller, 1981). His pioneering experience was bad understood at the time. It opened up a very new door with many philosophical and challenging consequences. Hiller relates the first reactions:

> *The first time I gave a talk on the subject of computer music was in 1956 before an audience of about 2000 computer experts and engineers at a Los Angeles meeting of the Association of Computing Machinery. Attitudes toward this early work ranged from curious to skeptical to overtly hostile. Rather interestingly, computer scientists were more open minded than musicians, and musicians were more open minded than scholars in the humanities, many of whom seemed to regard me as monstrous.* (Hiller, 1981)

His article on the *Illiac Suite* for *Scientific American* led to the hysterical attention of the popular press, and a storm of controversy that did not subside until computer music became available to everyone in the late 1980s. So virulent was the hostility of the musical establishment against this scientific poacher in the realms of art that both Baker's *Biographical Dictionary of Musicians* and the *New Grove Dictionary of Music and Musicians* refused to recognize his existence until just before his death –even though he had become internationally famous and was performed worldwide. (Kallisti Music Press)

The research and development of algorithmic music composition systems has several benefits. They have become more and more clear as musicians and researchers worked since Hiller's start up. In his paper *Composition Processes* (Koenig, 1978), Gottfried Michael Koenig distinguishes three main computer uses for purposes of composition. According to him, they are more likely to be used:

1- to solve parts of problems or to compose shorter formal sections instead of complete pieces,

2- to try out models greatly simplifying compositional reality and supplying the composer with a basic scheme which he can elaborate as he feels best,

3- to compose an individual piece for which the composer writes a special program more resembling a score than a solution for a number of problems.

My experience completely agrees with the Koenig's perception. From my point of view, the main benefits of algorithmic music composition systems are the following (Hinojosa, 2003):

1- These systems stimulate the composer's creative imagination in a very new and promising way, with lots of possibilities.

2- Composition programs can handle much more data and much faster than a human composer. They let him think in a high level of abstraction, leaving low-level details to the computer.

3- They are a door for searching new aesthetic concepts, new sonic conceptions and new ways of organizing sounds. So, they are a path for music development.

4- These systems allow scientific verification of music theories, when it is intended to simulate a known musical style in order to analyse and study it.

5- They allow to better know how musical processes take place in the human mind, so they let us know better the nature of the human being.

## 4.12 Conclusions

In this chapter I have discussed some aspects around Active Instruments, trying to give partial answers to some of the questions I intend to identify and address for achieving the objective of this research work. I have also exposed some relevant definitions that together with the definitions aforementioned in Chapter 1 (1.1), and a classification according to Robert Rowe's one (Rowe, 1993), give a conceptual framework for a better understanding of the nature of the problem. In the next chapter I will propose some possible future directions for enhancing the given partial answers, for addressing all the questions and eventually, for achieving the objective.

# Chapter 5

# Future Work and Conclusions

For deepen into the nature of the problem and achieve the objective of this research work, I could carry out some strategies through the study of several models. Additionally, I should read and study the three P.h.D. dissertations aforementioned (Bidlack, 1990; Manzolli, 1993; Dabby, 1995), relevant to the application of chaotic systems for the generation (or variation) of musical materials. My objective is to investigate and learn how these research have been carried out, what could they contribute to my research, and what my research could contribute to the field of Fractal Music. Finally, I could compare and contrast my general results with state-of-the-art relevant on-going research, and elaborate the final conclusions. The possible models to study follow:

## 5.1 Study of the model: Ifs + Simple Mapping + Mathematical GUI

To my mind, the simplest model involves an iterated chaotic function, a simple mapping strategy, and a mathematical graphical user interface. A complex mapping includes elements of musical knowledge (concepts, notions…) and could be very tricky, while simple mapping remains less musical, and algorithmically clear and direct. A mathematical GUI shows details of the state of the inner chaotic model; that is, the values of the involve variables, and lets the user change some or all these values interactively in realtime. This model could be studied in the context of several experimental software prototypes.

At the moment of writing this I have just made some experiments which gave me very encouraging results. I programmed an extern PD module which implements a Henon map (see Chapter 3). Later I made a PD program where the GUI shows two internal model parameters: *Angle* and *Orbit*; they can be changed interactively. The used mapping strategy was simple. Nevertheless, I intend to achieve satisfactory aesthetic results, musical logic and coherence. Thus, some simple but useful ideas were implemented in order to achieve **analogy** and **contrast** in the generated one-voice pitch sequence.

According to my previous experience and the suggestions of a composer who worked with me (Fernando Rodríguez Alpízar), an algorithmic music system not-inspired on any known music style should intend to achieve both properties as the most basic strategy to achieve satisfactory aesthetic results, musical logic and coherence. These features were successfully achieved in the obtained sonic results. Thus, our theoretical presumption on analogy and contrast was strengthened.

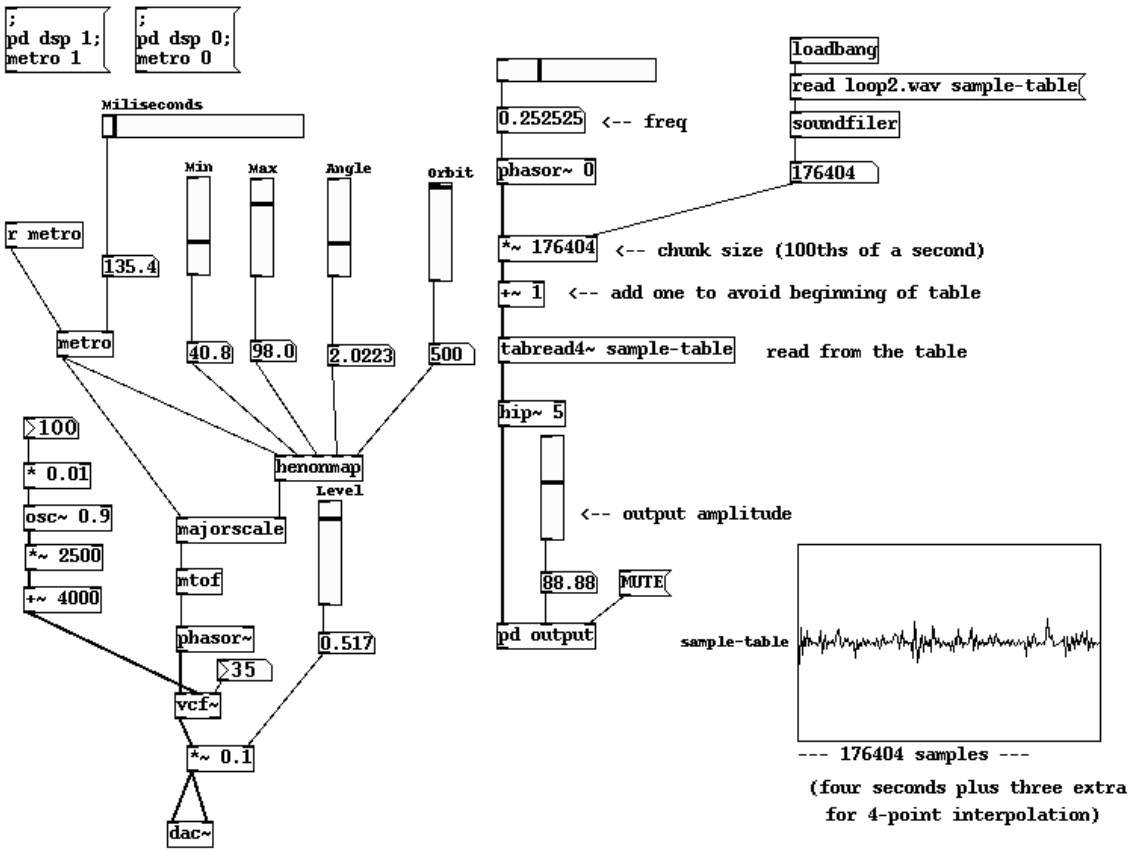



**Figure 5.1** PD experimental program.

For the study of this model (Ifs + Simple Mapping + Mathematical GUI) and achieve the goals, I propose the following methodology:

1- Design of relevant algorithms.
2- Development of several experimental software prototypes which use simple mapping and mathematical GUI. They will be able of generating simple musical textures.
3- Test the validity and viability of the theoretical principles and the usefulness of the software based on the usage by amateur and professional musicians.
4- Interviews with the involved individuals in order to get feedback.
5- Evaluation of results.

## 5.2 Study of the model: Ifs + Complex Mapping + Musical GUI

This model could be studied in the context of my system *Fractal Composer* (see Chapter 3), that uses complex mapping strategies and musical GUI. It has several features that obviously are relevant to the whole research. In fact, it is a tangible example of an Active Instrument based on chaotic functions. For the study of this model and achieve the goals, I propose the following methodology:

1- Reevaluate our previous results and compare it with similar projects.
2- Continue our previous work by testing the validity and viability of the theoretical principles and the usefulness of the software, based on the usage by more amateur and professional musicians.
3- Interviews with new involved individuals in order to get feedback.
4- Evaluation of results.

## 5.3 Study of the model: Ifs + Simple and Complex Mapping + TUI

In the frame of the MTG interactive group we work now (Septembre 2003) on a project named *reacTable**, which encompasses some theoretical concepts presented in MAX (Pukette, 1990) and FMOL (Jordà, 2002a). The original idea actually comes from Jordà's experience in designing instruments, in making music with them, and in listening and watching the way other people have played them. It activates important interdisciplinary research in the field of Computer Music, that significantly departs from the MTG traditional work based on signal processing techniques. Involved in this project are some research areas like algorithmic composition and realtime music creation / composition.

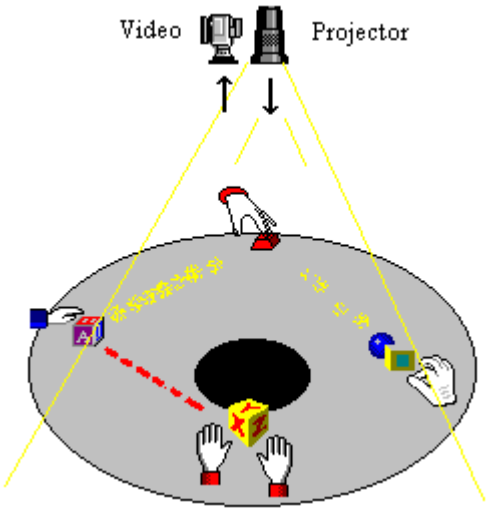



**Figure 5.2** The *reacTable** simplified scheme.

The *reacTable** is "*a table-based collaborative music instrument that uses computer vision and tangible user interfaces technologies, within a MAX-like architecture and scheduler, and with FMOL-inspired HCI models and visual feedback*" (Jordà, 2003). It is conceived both as a musical instrument and as an interactive musical installation, an installation for collaborative music creation in real time (i.e., a collective musical instrument). Several plastic or wood objects are placed on the table, which represent sound generators, controllers, etc., like in the MAX graphical programming environment. Over the table a video camera "looks" at the surface, sensing what is happening on it, for instance, what objects are placed, its positions, colors and any other feature useful to be used as a source of control. At the same time, a projector shows on the table a totally dynamic and interactive interface.

The projection follows the objects on the table wrapping them with auras or drawing figures on top of them, as well as covers the whole table surface with dynamic and abstract elements which reflect all the system's activity, and depend on the movements and trajectories of the performer's hands, the type and position of each object, and the relationships between all of them. The projected image never shows buttons, sliders or widgets of any kind. (Jordà, 2003)

The *reacTable** should be collaborative (on-line and off-line) and as intuitive as possible, so it won't need written instructions. It would neither has any mouse, nor keyboard, cables, nor wearables, but should be playable from the first second. It should be totally controllable, so no random or hidden presets will be present, thought some objects include automatic compositions algorithms which will have a degree of unpredictability and randomness. It should also be sonically interesting. This project actually intends to represent the state-of-the-art among its similars.

The learning curve is one of the challenging aims of the project. Although it should be playable from the first second, it should admit an almost unlimited learning (months, years…). So, it should be suitable and appealing for complete novices or amateur musicians, as well as for advanced electronic or professional musicians. The *reacTable** should allow a reasonable number of users, and they should be able to enter or leave the instrument-installation without previous announcements.

The *reacTable** will have, like in MAX, several kinds of objects such as Control Objects and Sound Generation Objects, as well as different types of connections between them, i.e., control and sound connections. These types will be reflected by de graphic design of the projected interface, which will draw these connections in different ways, as control or sound flows. Users will also be able to interact with the installation-instrument by means of their hands. For instance, interrupting a control flow would be possible by hitting the projected connection (or control flow) on the table, with an open hand in a similar way as a karateca does.

The *reacTable** project has started in December 2002 coinciding with the foundation of the Interactive Sonic Systems within the Music Technology Group, and since then we have been developing the basic ideas that support it. Now we work in parallel on the research of the main involved areas, i.e., computer vision and objects recognition, sound engine architecture, interactivity logic, sound visualization, interactive realtime algorithmic music generation, etc., while simultaneously designing the core and the integration of all these branches. Derived from my previous related work and interest, I propose the development and implementation, inside the *reacTable** project, of the followings ideas:

1- Design and development of an interactive algorithmic music system
2- for realtime performances
3- not-inspired (no mimic) on any known musical style
4- which could act as an active (self-regulated) instrument
5- "*where the user's actions would serve to influence an ongoing musical output rather than have the task of initiating each sound*" (Bischoff, 1991).
6- The core of the algorithms for music generation would be based on functions taken from the chaos and fractal theories.

The main idea would be the design and development of Control Objects which encapsulate the above listed points. These objects could not only be used for controlling Sound Generation Objects, but any other kind of objects, because they simply generate numbers which can be interpreted in different manners, such as MIDI data or general control data. A simplified scheme could be the following:

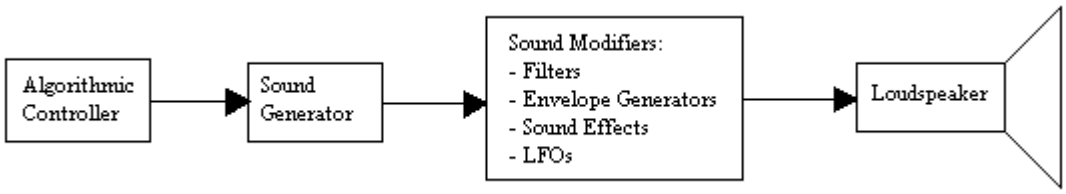



**Figure 5.3** Simplified scheme of *reacTable** object linking.

At this moment we have a working virtual prototype, developed by Geiger & Kaltenbrunner on PD and Java, which has been a very useful tool for the development and testing of several ideas we are working on. It covers a wide range of problems that we should solve on the real prototype, and makes up a good software environment for testing and assessing my generative chaotic algorithms.

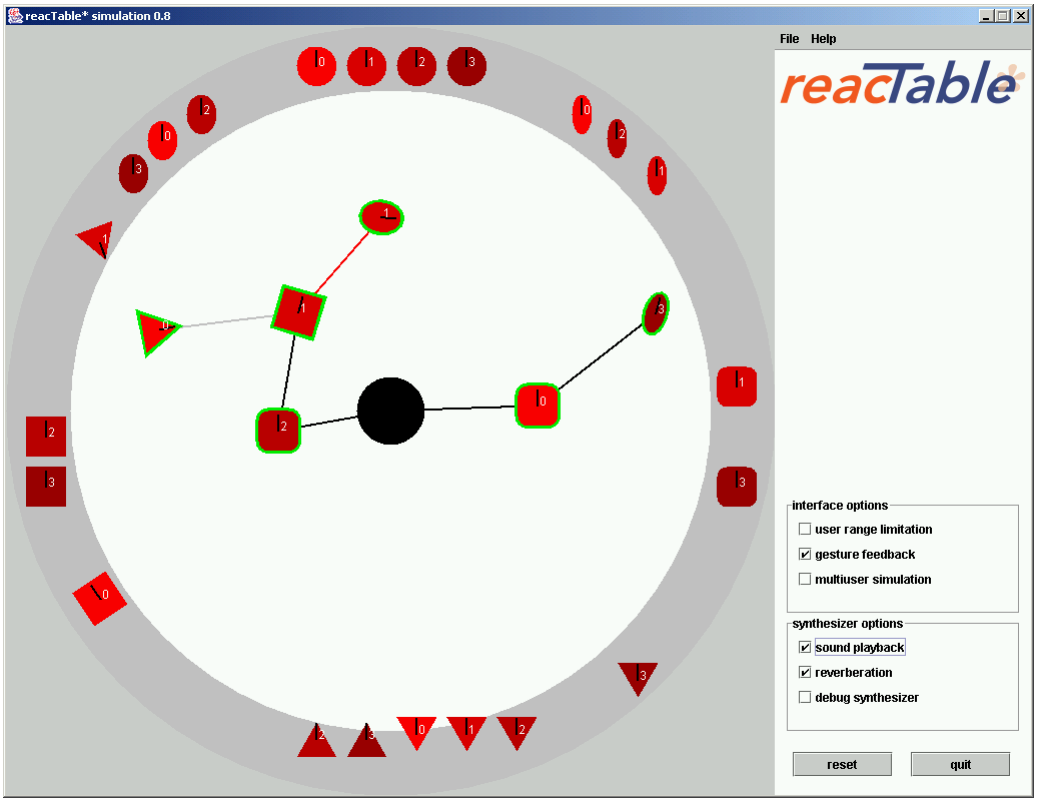



**Figure 5.4** Snapshot of the *reacTable** virtual simulator (by M. Kaltenbrunner & G. Geiger).

For the study of this model (Ifs + Simple and Complex Mapping + TUI) and achieve the goals, I propose the following methodology:

1- Design of relevant algorithms.
2- Development of self-regulated Control Objects and testing of their usefulness through several software prototypes, like PD experimental programs or the Geiger & Kaltenbrunner *reacTable** virtual simulator.
3- Insertion of my generative algorithms (Control Objects) in the real instrument prototype. There should be established a connection between the software models and its material representation in the real world (Tangible User Interface).
4- Analysis of the experimental applications results based on the usage by amateur and professional musicians.

## 5.4 Study of the model: (seed pattern + Ifs) + Complex Mapping + Musical GUI

In her paper *Musical variations from a chaotic mapping* (Dabby, 1995, 1996), MIT researcher Diana S. Dabby proposes a technique for generating musical variations of an original work from a chaotic mapping. Although this approach explores another perspective, variation versus generation, I think that could be relevant to this research. Thus, another future direction could be the study of this model in the context of an experimental software prototype, under the assumption that a complex mapping plus a musical GUI would lead to better practical results.

For the study of this model and achieve the goals, I propose the following methodology:

1- Design of relevant algorithms.
2- Development of an active instrument software prototype which accept as input a sequence of pitches and, after that, be able of generating musical variations from the initial sequence in a complex musical texture.
3- Test the validity and viability of the theoretical principles and the usefulness of the software, through the creation of musical works and performances by amateur and professional musicians.
4- Interviews with the involved individuals in order to get feedback.
5- Evaluation of results.

## 5.5 Conclusions

Research on algorithmic music is a fascinating and still open field. There are still a lot of things to do in order to improve the techniques and ideas that have been developed since algorithmic music with computers was born. Specifically the development of Active Instruments (*realtime algorithmic music systems* plus *interactive control*) for composition / improvisation / performance, seems to be a still under-explored field, as Sergi Jordà remarks:

> *Improvisation using computers still seems a burgeoning and under explored multidisciplinary area where the design of new controller interfaces, real-time sound synthesis and processing techniques, music theory, cognitive science, algorithmic composition techniques, and existing models of improvisation (computer and non-computer based) can converge, in order to bring new interactive music-making paradigms.* (Jordà, 2002b)

I have presented an overview of my research work concerned with the possible development of a new model of computer music system: Active Musical Instruments. I showed that the basis of this model is present in several ideas that have emerged in the recent past. I also showed that here converge concepts from relevant research areas that are under continuous and growing development, such as: Algorithmic Music Systems, Interactive Music Systems, Mapping, and Human Computer Interfaces. I intend to contribute to the setting-up of a formalized framework for the study of Active Instruments as a model of a new kind of computer-based musical instrument. So, the elaboration of a theoretical, conceptual and methodological framework for the study, design and development of these instruments was proposed.

The generative nature of an active instrument imply a generative model for its core. I explained why iterated and chaotic functions could be a good candidate for the development of this generative mathematical core. Thus, I also proposed the study of this kind of model inside the proposed framework as a particular and narrow approach to the nature of the problem. Finally, I suggested some possible future directions for achieving the objectives.

# Bibliography


Ames, Ch. 1982. “Crystals: Recursive Structures in Automated Composition”. *Computer Music Journal* 6, no. 3 (Fall 1982): 46-64.

Andrews, R. 1997. “Center for New Music & Audio Technologies (CNMAT): Studio Report”. In *Proceedings of the 1997 International Computer Music Conference*, Thessaloniki, Greece, Sept. 1997. http://cnmat.cnmat.berkeley.edu/ICMC97/papers-html/StudioReport.html

Assay, G. 1993. “Cao: vers la partition potentielle”. In Les Cahiers de l'Ircam: *La composition assistée par ordinateur* 1(3), juin 1993.

Barron, L. & B. 1989. Original MGM soundtrack from the film *Forbidden Planet*. Beverly Hills: Small Planet cop.

Belkin, Alan. *A Practical Guide to Musical Composition.* http://www.musique.umontreal.ca/personnel/Belkin/bk/4a.html#2)%20Contrast

Bidlack, R. 1990. “Music from Chaos: Nonlinear Dynamical Systems as Generators of Musical Materials”, Ph. D. dissertation, University of California, San Diego, 1990.

Bidlack, R. 1992. “Chaotic Systems as Simple (but Complex) Compositional Algorithms”. *Computer Music Journal* 16, no. 3 (Fall 1992): 33-47.

Bischoff, J. & Perkis, T. 1990. *Artificial Horizon*. CD booklet, Artifact Recordings.

Bischoff, J. 1991. “Software As Sculpture: Creating Music From the Ground Up”. Leonardo Music Journal, Vol. 1 No. 1, 1991.

Bolognesi, T. 1983. “Automatic Composition: Experiments with Self-similar Music”. *Computer Music Journal* 7(1), pp 25-36, Massachusetts Institute of Technology, 1983.

Brncic, G. 2002. *Ronde Bosse. Interactions électroacoustiques en temps reel*. GRM, Les concerts Multiphonies 2002. http://www.iua.upf.es/mtg/pdf/gbrncic.pdf. http://www.ina.fr/grm/agenda/multiphonies.fr.html.

Chadabe, J. 1984. “Interactive Composing: An Overview”. In *Computer Music Journal*, 8(1), 1984. Reprinted in Roads, C. (Ed.), *The Music Machine.* Cambridge: The MIT Press, 1989.

Chadabe, J. 1997a. *Electric Sound: The Past and Promise of Electronic Music*. Prentice Hall, New Jersey.

Chadabe, J., Richard, L., Zicarelli, D. 1997b. *M: The Intelligent Composing and Performing System*. User Manual, version 2.5 – August 1997. http://www.synthesisters.com/download/M.pdf

Chadabe, J. 2001. "The Multimedia Instrument". MIM (Laboratoire Musique et Informatique de Marseille), colloque Intersens, Marseille – 2000. http://www.labo-mim.org/pdf%20intersens/Chadabe.pdf

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.

Cope, D. 1991. *Computers and Musical Style*. Madison, WI: A-R Editions, 1991.

Cope, D. 1996. *Experiments in Musical Intelligence*. Madison, WI: A-R Editions, 1996.

Cope, D. 2000. *The Algorithmic Composer*. Madison, WI: A-R Editions, 2000.

Cope, D. 2001. *Virtual Music: Computer Synthesis of Musical Style*. Cambridge: The MIT Press, 2001.

Dabby, D. 1995. "Musical Variations from a Chaotic Mapping". Ph. D. dissertation, Massachusetts Institute of Technology, 1995.

Dabby, D. S. 1996. "Musical variations from a chaotic mapping". *CHAOS* 6 (2), American Institute of Physics, 1996.

Dapoigny, R., Benoit, E., Foulloy, L. 2003. "Agent-Based Implementation on Intelligent Instruments". In *Proceeding of 16th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2003*, pp 505-515, Laughborough, UK, June 23-26, 2003. Chung et al. (Eds.): *Developments in Applied Artificial Intelligence*, Lecture Notes in Computer Science 2718 Springer-Verlag 2003.

Degazio, B. 1993. "Towards a Chaotic Musical Instrument". In *Proceedings of ICMC*, Tokyo, pp393-395, International Computer Music Association, San Francisco, 1993.

Di Scipio, A. 1990. "Composition by Exploration of Nonlinear Dynamical Systems". In *Proceedings of the 1990 International Computer Music Conference*. International Computer Music Association: San Francisco, pp. 324-327, 1990.

Dodge, Ch. & Bahn, C. R. 1986. "Musical Fractals: Mathematical Formulas Can Produce Musical as well as Graphic Fractals". *Byte*, June 1986, pp. 185-196.

Dodge, Ch. 1988. "Profile: A Musical Fractal". *Computer Music Journal* 12, no. 3 (Fall 1988): 10-14.

Dodge, Ch., Jerse, T. A. 1997. *Computer Music: Synthesis, Composition and Performance.* Second Edition. New York: Schirmer Books.

Eaglestone, B. (ed), Brown, G., Ford, N., Moore, A., Nuhn, R. 2002. *Requirements Specification for a Composition Tools System.* University of Sheffield, UK, 2002. http://www.diku.dk/musinf/mosart/midterm/T6%20First%20Deliverable%20Vf.pdf

Eli, V. 1998. *ROBERTO VALERA. Catálogo de Compositores*. Fundación Autor, Madrid.

Fagarazzi, B. 1988. "Self Similar Hierarchical Processes for Formal and Timbral Control in Composition". *Journal of New Music Research*, 17, pp45-61, 1988.

Fischman, R. 2003. "Clouds, Pyramids, and Diamonds: Applying Schrödinger's Equation to Granular Synthesis and Compositional Structure". *Computer Music Journal*, 27:2, pp. 47–69, Summer 2003, Massachusetts Institute of Technology.

Gardner, M. 1978. "Mathematical Games: White and brown music, fractal curves and one-over-f fluctuations". *Scientific American*, 4, 16-32.

Gogins, M. 1991. "Iterated Functions Systems Music". *Computer Music Journal* 15, no. 1 (Spring 1991): 40-48.

Hiller, L., Isaacson, L. 1959. *Experimental Music: Composition with an Electronic Computer*. New York, Toronto, London: McGraw-Hill Book Company, Inc. Reprint, Westport, Conn.: Greenwood Press, 1979.

Hiller, L., Baker, R. A. 1964. "Computer Cantata: A Study in Compositional Method". *Perspectives of New Music,* 3 (Fall-Winter 1964): 62-90.

Hiller, L. 1981. "Composing with Computers: A Progress Report". *Computer Music Journal* 5, no. 4 (winter 1981): 7-21. Reprint in *The music machine*, ed. Curtis Roads, 75-89. Cambridge, Mass.: MIT Press, 1989.

Hiller, L. 1989. *Computer Music Retrospective (1957 – 1985)*. CD booklet, Wergo, Germany.

Hinojosa, R. 1993. *Sistemas de Ayuda a la Composición Musical*. Thesis work, Faculty of Mathematics and Computer Sciences, University of Havana.

Hinojosa, R. 1994. "Sistemas de Ayuda a la Composición Musical: la Experiencia del EMEC". *Proceedings of the I International Congress of Informatics on Culture*, International Convention INFORMATICA '94, Havana, February 1994.

Hinojosa, R. 1995. *Acerca de los Instrumentos Electrónicos, la Música Electroacústica y las Computadoras*. Havana: Banco de Ideas Z / Instituto Superior de Arte.

Hinojosa, R. & Cabrera, C. 1997. *La Música Electroacústica: Uno de los Caminos de la Música. Conversación con Roberto Valera*. Havana, 1997. Unpublished.

Hinojosa, R. & Cabrera, C. 1998. *La Música Electroacústica Puede Ser Tan Fuerte y Tan Humana Como Cualquier Otra Música. Conversación con Adolfo Nuñez*. Havana, 1998. Unpublished.

Hinojosa, R. 1999. *Fractal Composer*. User Manual, version 1.0 – EMEC/ISA, Havana, 1999.

Hinojosa, R. 2000. "Fractal Composer: Un Instrumento del Siglo XXI". *Proceedings of the IV International Congress of Informatics on Culture*, International Convention INFORMATICA 2000, Havana, May 2000.

Hinojosa, R. 2001. *Music Generation Panel (A critical review)*. MOSART Workshop on Current Research Directions in Computer Music, Barcelona, Nov 15-16-17, 2001, Institut Universitari de l'Audiovisual, Universitat Pompeu Fabra. http://www.iua.upf.es/mtg/mosart/panels/music-generation.pdf

Hinojosa, R. 2003. "Some Projects and Reflections on Algorithmic Music". *Proceedings of Computer Music Modeling and Retrieval 2003*, Montpellier, France. Reprinted in Uffe Kock Wiil (ed.), Springer Verlag - Lecture Notes in Computer Science (LNCS) Series Vol. 2771.

Jacob, B. 1996. "Algorithmic Composition as a Model of Creativity". *Organised Sound*, volume 1, number 3, December 1996. http://www.ee.umd.edu/~blj/algorithmic_composition/algorithmicmodel.html

Jordà, S. 2002a. "FMOL: Toward User-Friendly, Sophisticated New Musical Instruments." *Computer Music Journal*, Vol. 26, No.3. pp 23-39, 2002.

Jordà, S. 2002b. "Improvising with Computers: A Personal Survey (1989-2001)". *Journal of New Music Research* Vol.31 .1 pp 1-10.

Jordà, S. 2003. "Sonigraphical Instruments: From FMOL to the reacTable*". *Proceedings of 2003 International Conference on New Interfaces for Musical Expression*. Montreal, Canada.

Kellert, S. 1993. *In the Wake of Chaos*. The University of Chicago Press.

Koenig, G. M. 1978. "Composition Processes". *Computer Music*, M. Battier & B. Truax, eds. Canadian Commission for Unesco, 1980. Original republished in Gottfried Michael Koening, *Aesthetische Praxis/Texte zur Musik*, Band 3, 1968-1991, PFAU Verlag, 1993, pp. 191-210, as Kompositionsprozesse. http://academic.evergreen.edu/a/arunc/compmusic/koenig/koenig.pdf

Kondrátov, A. 1989. *El intelecto electrónico*. Moscú: Mir.

Little, D. 1991. "Composing with Chaos: Applications of a New Science for Music". *Journal of New Music Research*, 22(1), pp23-51, 1991.

Lovelace, A. 1842. Ada's Notes found in *Ada, The Enchantress of Numbers* by Betty Alexandra Toole Ed.D., Strawberry Press, Mill Valley, CA 1992. Revised Paperback Edition 1998. The original Memoir was printed in *Scientific Memoirs, Selections from The Transactions of Foreign Academies and Learned Societies and from Foreign Journals*, edited by Richard Taylor, F.S.A.,Vol III London: 1843, Article XXIX. Sketch

of the Analytical Engine invented by Charles Babbage Esq. By L. F. Menabrea, of Turin, Officer of the Military Engineers. [From the Bibliothque Universelle de Gnve, No. 82 October 1842]. http://www.agnesscott.edu/lriddle/women/ada-love.htm

Loy, G. 1988. "Composing with Computers - a Survey of Some Compositional Formalisms and Music Programming Languages". In M. Mathews & J. R. Pierce, eds. *Current Directions in Computer Music Research*. Cambridge, Massachusetts: The MIT Press. pp 291-396.

Mandelbrot, B. 1975. *Les Objets Fractals*. Editions Flammarion, Paris.

Mandelbrot, B. 1983. *The Fractal Geometry of Nature*. Freeman: San Francisco.

Manzolli, J. 1993. "Nonlinear Dynamics and Fractals as a Model for Sound Synthesis and Real-time Composition", Ph. D. dissertation, University of Nottingham, 1993.

Miller, J. 1985. "Personal Composer". In *Computer Music Journal*, 9 (4), 27-37. Reprinted in Roads, C. (Ed.), *The Music Machine.* Cambridge: The MIT Press, 1989.

Miranda, E. R. 2001. *Composing Music with Computers*. Oxford, UK: Focal Press.

Moles, A. 1960. *Les Musiques Expérimentales*. Editions du Cercle d'Art Contemporain.

Monro, G. 1993. "Synthesis from Attractors". *Proceedings of ICMC*, Tokyo, pp390-392, International Computer Music Association, San Francisco, 1993.

Nagashima, Y., Katayose, H., Inokuchi, S. 1993. "Deterministic Chaos, Iterative Models, Dynamical Systems and Their Application in Algorithmic Composition". In *Proceedings of ICMC*, Tokyo, pp194-197, International Computer Music Association, San Francisco, 1993.

Nettheim, N. 1992. "On the Spectral Analysis of Melody" *Journal of New Music Research*, Vol 21, 1992, pp. 135-148.

Pachet, F. 2000. "Qu'est ce qu'une mélodie intéressante?". *La Recherche*, November 2000. Hors Série "Les origines de l'Art".

Papadopoulo, G., Wiggins, G. 1999. "AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects". In *Proceedings of the AISB'99 Symposium on Musical Creativity*.
http://www.soi.city.ac.uk/~geraint/papers/AISB99b.pdf

Polotti, P. & Evangelista, G.2001. "Fractal Additive Synthesis via Harmonic-Band Wavelets". *Computer Music Journal*, 25:3, pp. 22–37, Fall 2001

Pukette, M. & Zicarelli, D. 1990. *MAX : An Interactive Graphical Programming Environment*. Menlo Park : Opcode Systems.

Risset, J.-C. 1986. “Arte e Scienza: musica elettroacustica numerica”. In *Nuova Atlantide: il continente della musica elettronica 1900-1986* - catalogo della mostra della Biennale di Venezia 1986, pp. 102-120

Roads, C. 1986. “Ricerche sulla musica e l’Intelligenza Artificiale”. In *Nuova Atlantide: il continente della musica elettronica 1900-1986* - catalogo della mostra della Biennale di Venezia 1986.

Roads, C. (ed) 1989. *The Music Machine.* Cambridge, Massachusetts: MIT Press.

Roads, C. 1996. *The Computer Music Tutorial*. The MIT Press, Second printing.

Robledo, E., Hinojosa, R., de Boer, M. 2002. “A C++ Development platform for Real-time audio processing and synthesis applications”. *Proceedings of the 5th International Conference on Digital Audio Effects*. Hamburg, Germany, 2002.

Rolnick, N. 1992. “Performing with Active Instruments”. In Letters, *Computer Music Journal.* Cambridge, Massachusetts: MIT Press. 16:3 1992.

Rosenboom, D. 1992. “Interactive Music with Intelligent Instruments – A New, Propositional Music?”. Brooks, I.(ed.): *New Music Accross America*, California Institute of the Arts in conjunction with High Performance Books, Valencia and Santa Monica, CA, 1992. http://music.calarts.edu/~david/writings/NMAA.html

Rowe, R. 1993. *Interactive Music Systems: Music Listening and Composing.* Cambridge, Mass.: The MIT Press.

Schloss, A. & Jaffe, D. 1993. “Intelligent Musical Instruments: The Future of Musical Performance or the Demise of the Performer?”. *INTERFACE Journal for New Music Research*, The Netherlands, December 1993. http://www.finearts.uvic.ca/~aschloss/Articles/INTERFACEarticle.html

Siegel, W. 1994. “Do androids dance to computer music?”. *Nordic Sounds*, No. 1, 1994.

Spiegel, L. 1987. “A Short History of Intelligent Instruments”. A followup to material recently published by CMJ. Letter to the Editor, *Computer Music Journal*, Vol. 11, #3, Fall, 1987. http://retiary.org/ls/writings/cmj_intelligt_instr_hist.html

Spiegel, L. 1992. “An Alternative to a Standard Taxonomy for Electronic and Computer Instruments”. Commentary written at the request of the Editor of Computer Music Journal, but published as a Letter to the Editor, *Computer Music Journal*, MIT Press, Vol. 16, #3, Fall 1992, pp.5-6. http://retiary.org/ls/writings/cmj_taxonomy.html

Spiegel, L. 1998a. “Graphical GROOVE: memorial for the VAMPIRE, a visual music system”. *Organised Sound*, Cambridge University Press, 3(3): 1998, pp.187-191. http://retiary.org/ls/writings/vampire.html

Spiegel, L. 1998b. “An Information Theory Based Compositional Model”. *Leonardo Music Journal*, MIT Press, Volume. 7, January 1998, pp. 89-90. http://retiary.org/ls/writings/info_theory_music.html

Truax, B. 1990. “Chaotic Non-Linear Systems and Digital Synthesis: An Exploratory Study”. *Proceedings of ICMC*, Glasgow, pp100-103, International Computer Music Association, San Francisco, 1990.

Vanechkina, I. 1997. Journal Review on “In commemoration of R. Zaripov” appeared in a special issue of Artificial Intelligence News magazine, Moscow, Artificial Intelligence Association, 1995. Leonardo Electronic Almanac - Volume 5, Number 7, July 1997. http://mitpress2.mit.edu/e-journals/LEA/TEXT/lea5-7.txt

Vidal, T. 2000. *FRACTAL COMPOSER: El nuevo paradigma de la música*. Orbe, semanario internacional editado por Prensa Latina. Año 2, No. 2, semana del 10 al 16 de junio de 2000.

Voss, R.F. & Clarke, J. 1975. “1/f noise in music and speech”. *Nature*, 258:317-318.

Voss, R.F. & Clarke, J. 1978. “1/f noise in music: music from 1/f noise”. *Journal of Acoustical Society of America*, 63(1):258-263.

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.

Wessel, D. 1991. “Instruments That Learn”. *Computer Music Journal*, MIT Press, Vol. 15, No.4, Winter 1991.
http://cnmat.cnmat.berkeley.edu/News/Wessel/InstrumentsThatLearn.html

Winner, J. & Chusid, I. “Circle Machines and Sequencers”. Originally appeared in *Electronic Musician* magazine. http://raymondscott.com/em.html

Xenakis, I. 1986. “Dimensión Matemática de la Música”. *El Correo de la UNESCO*. Abril de 1986.

Xenakis, I. 1992. *Formalized Music: Thought and Mathematics in Composition*. Rev. ed. Stuyvesant, New York, Pendragon Press.

Yadegari, S. 1991. “Using Self-similarity for Sound/Music Synthesis". In *Proceedings of ICMC*, Montreal, pp423-424, International Computer Music Association, San Francisco, 1991.

Zaripov, R. 1971. *Cybernetics and music*. Moskva, Nauka.

Zaripov, R. 1979. *Musica con il calcolatore: le regole matematiche della composizione*. Franco Muzzio & C, Padova.

## Online References


Hiller, Lejaren.
Archive: http://ublib.buffalo.edu/libraries/units/music/spcoll/lhhome.html
Another Web Resources:
Lejaren Hiller Webpage: http://pw1.netcom.com/~kallisti/Hiller.html
University of Illinois Hiller Page: http://cmp-rs.music.uiuc.edu/history/hiller.html
Lejaren A. Hiller Webpage: http://chem.pdx.edu/~wamserc/Hiller/default.htm

Kallisti Music Press: http://pw1.netcom.com/~kallisti/Hiller.html

Martirano, Salvatore: http://ems.music.uiuc.edu/~martiran/HTdocs/bio.html

Music Technology Group (MTG): http://www.iua.upf.es/mtg/eng/

Pascal, Blaise 1660. *Pensées*. Translated by W. F. Trotter.
http://www.ccel.org/p/pascal/pensees/pensees.htm

Scott, Raymond. Official site: http://raymondscott.com/

Spiegel, Laurie. Official site: http://retiary.org/ls/

# Appendix 1

# Algorithmic Composition as a Constraint Satisfaction Problem

This is a final paper written after the lectures given by professor Dr. Héctor Geffner, in the doctoral course *Problem Solving in Artificial Intelligence* (2001-2002). Unpublished.

# ALGORITHMIC COMPOSITION AS A CONSTRAINT SATISFACTION PROBLEM

*Rubén Hinojosa Chapel*
Music Technology Group
Pompeu Fabra University
Ocata 1, 08003 Barcelona, Spain
+34 93 542 2104
ruben.hinojosa@iua.upf.es

**Abstract**

Throughout history, musicians have always used music composition rules, which actually are in most cases prohibitions or constraints. The elements of music with which composers work, as well as the possible combinations of those elements, make up an enormously big finite set of values. The composer's function is to select, from this big set, a subset of values, which will be arranged into a certain temporal organization, constrained by the music rules, and eventually create an artistic work. From a mathematical point of view, this process is closely related to one field of Artificial Intelligence: *Constraint Programming*. In this paper the author makes a reflection on the relationship between Music Composition and Constraint Programming. As a concluding idea, the author suggests that Constraint Programming could be a promising technique for the development of computer systems oriented to algorithmic composition. This text follows after the lectures given by professor Dr. Héctor Geffner, in the doctoral (2001-2002) course *Problem Solving in Artificial Intelligence*.

## 1 Introduction

*"Schönberg –has said someone– is an inventor of theorems, who always seems to be demonstrating it on the blackboard, with geometric formulae... What it is true is that music can never turn to forces of unconscious, as poetry or painting does; it is an art –though the term scare me– terribly Euclidian. And a Bach's fugue, as well as a motete by Victoria, can always be translated on the blackboard by means of geometric formulae."* [1]

The high grade of abstraction and formalism that music has on its theoretical aspects, has often lead to be compared to mathematics. Not without reason, one of the most important music disciplines, Harmony, is used to be raised to the category of a science. It has been stated, for instance, that musicians have invented the notion of coordinates before mathematicians did it. In fact, a music score is actually a coordinate system, where frequencies values are represented onto the ordinate axis, and the time flow is represented onto the axis of abscissas.

In his creative labour, the music composer works with the so-called elements of music, such as: melody, harmony, timbre, articulation, dynamics, form, etc. We could compare this process to the one followed by a cook who creates a new recipe. He has at hand a lot of ingredients that can be used, but he will only use a little portion. He will select a small portion from the set of ingredients that he can buy, for instance, in one or several

markets. After that, he will merge, prepare, cook, etc., these ingredients according to the rules and proportions studied and developed for centuries of culinary art. For example, in general, the amount of salt that he can use has a very low superior limit, because the food could become salty if he surpasses that limit.

Now think about a music composer who wants to create a new piece. He has at hand an enormously big set of values (of frequencies or pitches, durations or rhythms, intensities or dynamics, timbres or instruments, etc.) that he can combine horizontally as well as vertically, in lots, so many different ways. The combinatory explosion is enormous, which is confirmed by the high number of known music pieces. So, what criterion should be followed in order to select a subset of values that can be handled and feasible of being logically organized? When we say "being logically organized" we think on what is used to be called "musical logic" but, what defines a musical logic?

Ask a person without music knowledge, to seat at the piano and try to play some keys during a couple of minutes. Repeat later the same experiment with someone who owns music knowledge. Why do we think about anything but music when we listen to the first "player"? Why does the second playing will unmistakable sound, or will be recognized, as music? Answer: the first one lacks a system of rules, laws and constraints, while the second one owns such a system. The study and application of the rules, laws and constraints of music for centuries, has lead toward the creation of many sorted subsets of music elements and values, which we normally recognize as pieces of music.

Besides all the subjective factors which take part on the creation of a musical composition, among others: inspiration, intuition or experience, the musician selects objective "parameters" with which he designs, step by step, how will his piece of music be. Normally he will decide if he is going to use a polyphonic or homophonic texture, which will the structure be, for how many instruments, what kind of harmony, etc. In many cases he must have to keep himself inside a, more or less, strict set of constraints. For instance, he must not write a note for an instrument that cannot be played. Each instrument has a register or range of note values that can be played on it. In general, the study of music theory involves the study of numerous rules and restrictions. Let's see an example of a real musical rule, taken from a harmony book [2]:

> ***Linking of the tonic chord with the dominant and subdominant ones***
>
> *Before trying the most simple links of chords, it is necessary to take into account certain general* ***rules****:*
>
> *1- The common note to the chords we are trying to link (c-e-g  g-b-d) should remain on the same voice (...)*
> *2- All the voices but the bass, should be conducted as possible by joined grades, avoiding the frequently use of walking by disjoint grades or melodic jumps.*
> *3- The soprano and the bass voices should walk, with preference, by contrary movement.*
> *4- The distance between the bass and the tenor voices can surpasses sometimes the octave; but for obtaining a well balanced sonority it should be considered the octave as the maximum admissible distance between the soprano and the alto voices, and mainly, between the alto and the tenor voices.*

We have seen that the music composer, on his creation work, handles lots of variables, each of them has its own domain of values. These values are restricted by a set of rules, which should be simultaneously satisfied. But this situation we have just described is actually what it is known as a *Constraint Satisfaction Problem*, a kind of problem studied as a part of one of the Artificial Intelligent disciplines: *Constraint Programming.*

## 2 Constraint Programming

*Constraint Programming* (CP) is the study of computer systems based on restriccions. The main idea of CP lies on solving problems by means of constraint statements (requirements) related to the area of such problems, and therefore, finding the solutions which satisfy all the restrictions.

*"Constraint Programming represents one of the closest approaches computer science has yet made to the Holy Grail of programming: the user states the problem, the computer solves it."* (E. Freuder, cited in [3])

The representation of a problem by means of constraints is usually very flexible, because they can be added, deleted or modified. The programming process has two general steps:

1) generation of one representation of the original problem as a CSP (*Constraint Satisfaction Problem*), and
2) search of a solution to this CSP.

A CSP can be defined as:

- a finite set of *variables* $X = \{x_1,...,x_n\}$,
- for each variable $x_i$, a finite set $D_i$ of possible values (its *domain*),
- a finite set of *constraints* which restricts the values that can simultaneously be assigned to the variables over its respective domains.

A *solution to the CSP* is the assignment of a value to each variable, such that all the constraints are satisfied at the same time. A CSP is *consistent* if it has one or more solutions. When solving a CSP, we would wish to find:

- any solution, no matter which is,
- every possible solution,
- an optimal solution, or at least a good solution, according to a given objective function defined in terms of some or all variables.

The solutions to a CSP can be found by systematically searching through possible assignments of values for each variable. Searching methods are divided into two wide groups:

1) those which pass through the partial solution space (or partial values assignment),

2) and those which completely explore the space of values assignment in a stochastic way.

The task of constraints programming is to reformulate the initial problem as a CSP, and to solve it by means of general methods or related to a certain particular domain. General methods are usually related with techniques to reduce the search space, and with specific search methods. The fundamental idea is to convert a given CSP into an equivalent one; that is to say, into another CSP which owns the same solutions set, but easier to solve. This process is known as *Constraints Propagation*. Algorithms for constraints propagation reduce the search space and, therefore, restrict the combinatory explosion [4].

On the other hand, specific domain methods are usually provided as algorithms for specific purposes or specialized packs, often known as *Constraint Solver*s. Some examples are:

- a program that solves systems of linear equations
- a package for linear programming
- an implementation of the unification algorithm, a cornerstone of automated theorem proving [4].

We have seen that musicians have always been solving constraint satisfaction problems for creating their musical work, though without giving it that name. As a scientific discipline, Program Restriction has actually formalized and developed the theoretical-mathematical-algorithmic basis of one of the several intellectual processes that take place in the human mind, making possible its simulation (programming) by means of computer systems of Artificial Intelligence, and the resolution of real-life problems in an automatic way.

The evidence about the possibility of expressing music theories as a CSP, has lead some researchers toward the development of computer composition / harmonizing systems based on this formalism [5]. This theory has also been used for supporting automatic systems for musical analysis. In [6] the authors describe an interactive tool that uses constraint propagation inside an expert system for music composition, where traditional counterpoint is modeled as a CSP.

Another developed project has been implemented using a constraint programming language called Oz, and its name is COMPOzE. We would like to show, with this system, how we could apply the Constraint Satisfaction Problem theory, in practice, to algorithmic composition.

## 3 COMPOzE: Music Composition Through Constraint Programming

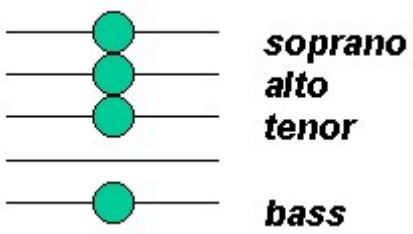
COMPOzE is an interactive system based on Constraint Programming, for the creation of simple four voices music pieces, giving a musical development scheme. This scheme (or plan) describes the harmonic flow and some features of the desired composition. The system shows a graphical user interface that lets the selection of musical rules through direct manipulation. The composition process, as well as the solutions, are represented

graphically. The user can also listen to the solutions and compare it. Eventually COMPOzE generates a MIDI file [7, 8].

The musical plan is made of a harmonic progression, for instance: **T**onic, **S**ubdominant, **D**ominant and **T**onic. It is also made of some parameters such as: tonality, tempo, rhythm, etc., and some features of the chords (major, minor, accidental notes, etc.). The chord sequence to be generated should obey the harmonic progression, as well as the programmed composition rules (constraints). These rules include restrictions for the notes (tessitura), for the chords (crossing prohibition of voices, bass distance), for the sequences (jump compensation, bass rhythms), etc. The following are real examples of these musical rules:

- **Crossing Prohibition:** The voices within one chord may not cross, in a sense that a lower voice may not play a higher note than a higher voice. For example, the bass may not play a higher note than the tenor within a chord.

- **Jump Law:** A jump of a voice from a chord to its neighbor that exceeds a given distance must be soothed in the following chord by a jump of one or two steps in the opposite direction.



**Chord**

In accordance with the initial hypothesis, the most suitable and natural framework for formalizing the composition of music is given by the constraint satisfaction theory. When representing this problem as a CSP, it would be expressed in the following way:

- In general we have $n$ x $v$ variables, where $n$ is the number of chords in the sequence, and $v$ the number of notes in each chord. In this case we have four voices by chord, namely: **B**ass, **T**enor, **A**lto and **S**oprano. We'll name the variables as follows: $B_i$, $T_i$, $A_i$, $S_i$, where $i \in \{1,\ldots, n\}$.
- The domain for these variables is given by a range of playable pitches.
- The harmonic functions and the composition rules can be formulated as constraints ruling between one or several variables. For example, the crossing prohibition can be expressed by the following constraint:
  $\forall i \in \{1,\ldots, n\}, B_i \leq T_i \leq A_i \leq S_i$

In a more formal way, the Constraint Satisfaction Problem could be expressed in the following terms:

**Variables and domains**

- 4 x $n$ variables $B_i$, $T_i$, $A_i$, $S_i$, $i \in \{1,\ldots, n\}$,
- with domain of integer values $\{0,\ldots,60\}$.

**Constraints:**

- Harmonic functions: $B_i \bmod 12 \in \{0, 4, 7\}$
- Tessitura: $\forall i \;\; 0 \leq B_i \leq 20$
- Bass distance: $\forall i \;\; B_i + BassDist \leq T_i, A_i, S_i$
- Jump compensation:

$$\forall i \;\; S_i - S_{i+1} \geq jump \rightarrow S_{i+2} - S_{i+1} \in \{1, 2\}$$
$$S_{i+1} - S_i \geq jump \rightarrow S_{i+1} - S_{i+2} \in \{1, 2\}$$

The goal of Constraint Propagation is to progressively restrict the set of possible values the variables can get, applying the restrictions until eventually only one value for each variable is found. The set of possible values is kept inside a structure called *constraint store*. For instance, the fact that the base pitch of the first chord must be taken from the first 25 pitches of the scale is expressed by the constraint: $B_1 \in \{0,\ldots,24\}$ in the constraint store. More complex constraints are expressed by propagators, which observe the constraint store and amplify it if possible.

A propagator inspects the store with respect to a fixed set of variables. When values are ruled out from the domain of one of these variables, it may add more information on others to the store, i.e., it may amplify the store by adding constraints to it. As an example consider the crossing prohibition. It can be expressed for the first chord by installing the following three propagators:

```
B1 =<: T1 T1 =<: A1 A1 =<: S1
```

To explain how they can amplify the constraint store, let us assume that

$A_1 \in \{30,\ldots,45\}$ and $S_1 \in \{25,\ldots,60\}$

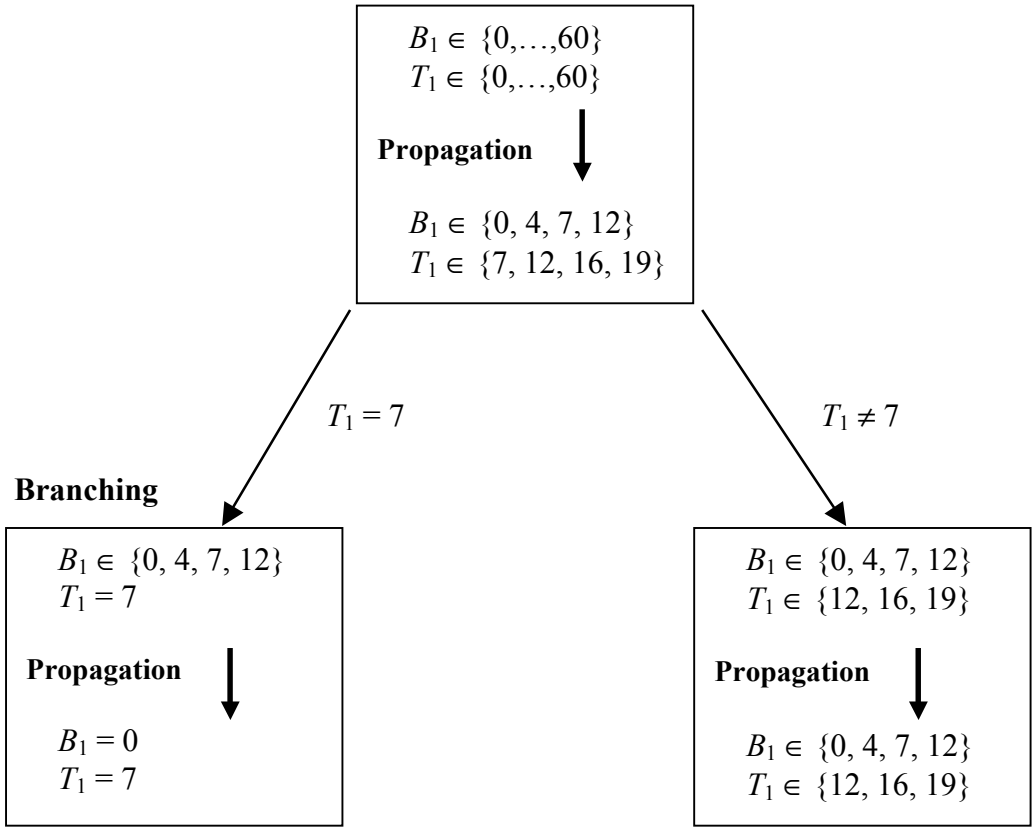
Then the third propagator will exclude the values 25,...,29 from the domain of $S_1$, reducing its domain to $\{30,\ldots,60\}$. Inversely, if we later know that $S_1 \in \{30,\ldots,40\}$, then $A_1$ will be restricted to the new domain $A_1 \in \{30,\ldots,40\}$. Note that this propagator remains active, waiting for more information on either $A_1$ or $S_1$. It only ceases to exist when it becomes clear that it will never amplify the store again. Let's see the following graphics:

**Initial constraints:**
$B_1 \in \{0,\ldots,60\}$
$T_1 \in \{0,\ldots,60\}$
$BassDist = 6$

**Propagation:**
$B_1, T_1 \bmod 12 \in \{0, 4, 7\}$
$0 \leq B_i, T_i \leq 20$
$B_1 + BassDist \leq T_1$

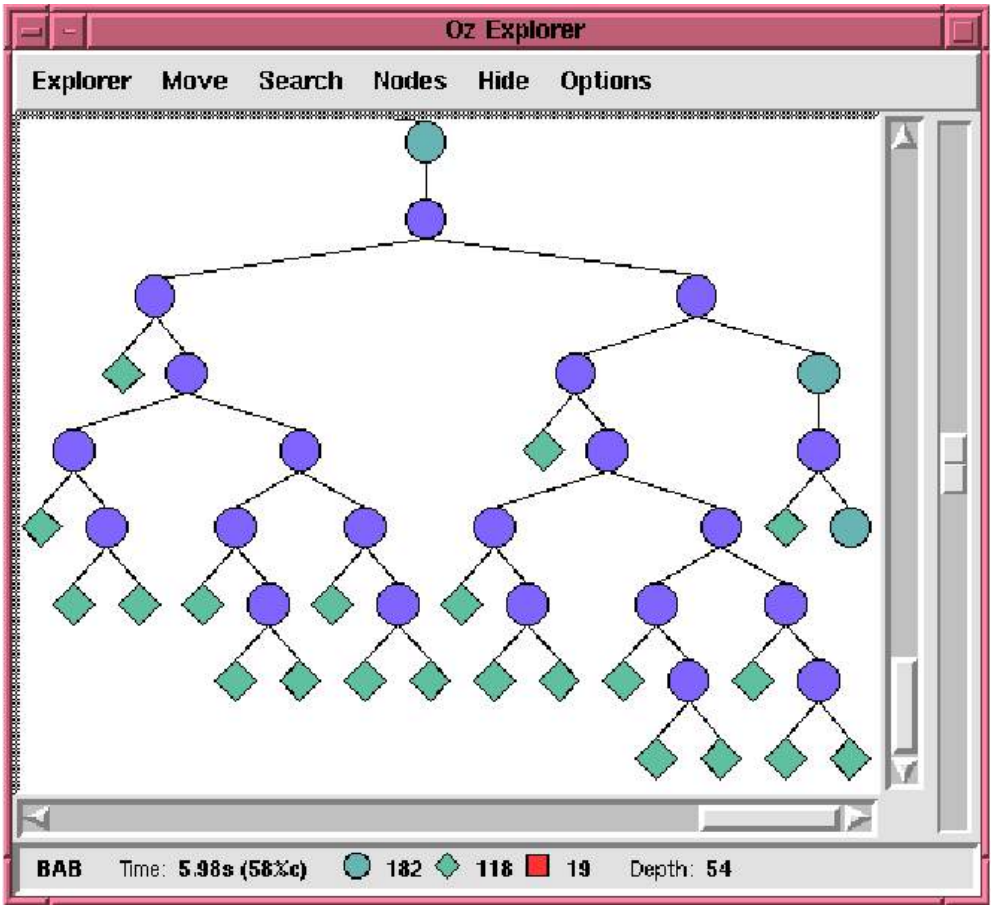
**Result of the propagation:**
$B_1 \in \{0, 4, 7, 12\}$
$T_1 \in \{7, 12, 16, 19\}$
$BassDist = 6$

**SEARCHING**

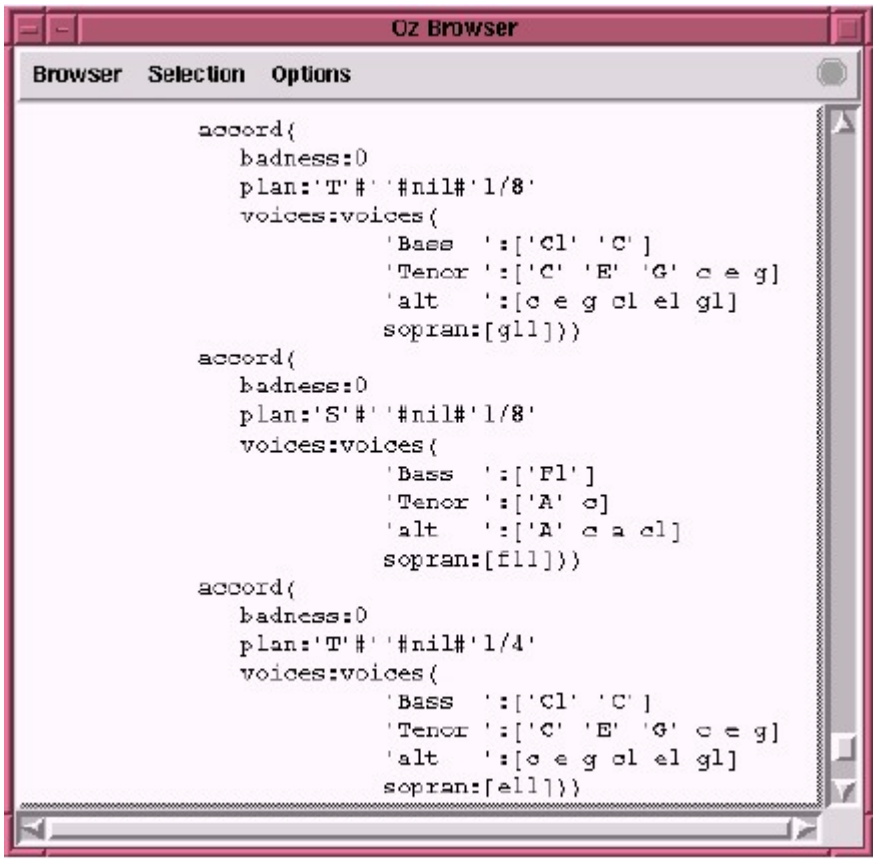



COMPOzE takes as input a musical plan, and gives as result one or more compositions, which are structured according to that plan. At the same time, they follow the defined user criteria. The system permits the user to decide, for each musical rule, if it should be ignored (`off`), strictly obeyed (`hard`), or preferable obeyed (`soft`), by means of a value (`weight`) between 0 and 100.

The implementation uses the *branch-and-bound* technique for minimizing the number of broken soft rules. If there are several of these rules, they are assessed by their weights. The Oz explorer shows the search tree while the user can, interactively, listen to the generated solutions by mouse click over the graphically represented solution nodes.




**The Oz explorer shows the search tree**




**A fragment of COMPOzE code inside the Oz browser**

## 4 CONCLUSIONS

Musicians have used throughout centuries of music practice, composition rules which actually are in most cases restrictions or constraints. They allow navigating and reducing an enormous search space formed by an enormously big finite set of musical values. This kind of problem is quite similar to those solved daily by everybody in many areas of the human activity. They have been studied and formalized by an Artificial Intelligence discipline: Constraint Programming.

The idea of music composition modeling as a constraint satisfaction problem emerges in a natural way, when we intend to develop algorithmic music systems with the tools Artificial Intelligence offers nowadays. On one hand, that vision is supported on the natural relationship between music composition and constraint programming, and on the other hand, on the different research projects carried out in the past years, not only for music composition, but also for harmonization or analysis.

In this paper we have discussed the existing relationship between an eminently intellectual artistic activity, music composition, and its possible scientific formalization with theoretical and practical objectives. With one of the developed projects, the COMOzE system, we have exemplified how this formalization could be carried out, as well as the automatic solution to the proposed problem; that is, the automatic creation of a simple piece of music.

Some interesting subjects, such as the discussion of philosophical topics around the necessity (or not) of creating automatic systems for music composition, are beyond the scope of this text. Nevertheless, we think conveniently to say that the study of this kind of problem can be as important, to Artificial Intelligence research, as the automatic solution of games like the 15 puzzle, the Rubik cube, or the *n* queens problem.

## 5 REFERENCES


1- Carpentier, Alejo. *Crónicas*. Tomo I, Editorial Arte y Literatura. La Habana, Cuba, 1975.
2- Scholz, Hans. *Compendio de Armonía*. Editorial Labor S.A., Barcelona, 1951.
3- Barták, Roman. *Constraint Programming: In Pursuit of the Holy Grail*. http://kti.ms.mff.cuni.cz/~bartak/downloads/WDS99.pdf
4- R. Apt, Krzysztof. *Constraint Programming*. ERCIM News No.39 - October 1999. http://www.ercim.org/publication/Ercim_News/enw39/apt.html
5- Pachet, F. and Roy, P. 2001. "Musical harmonization with constraints: A survey". *Constraints*, 6(1):7-19. 2001. http://www.csl.sony.fr/downloads/papers/2000/pachet-constraints2000.pdf
6- Ovans, Russell and Rod Davison. "An interactive Constraint-Based Expert Assistant for Music Composition". *Proceedings of the Ninth Canadian Conference on Artificial Inteligence*, pp. 76-81. http://citeseer.nj.nec.com/ovans92interactive.html
7- Henz, Martin, Stefan Lauer, and Detlev Zimmermann. "COMPOzE --- Intention-based Music Composition through Constraint Programming". *Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence*, Nov16--19 1996, IEEE Computer Society Press. ftp://ftp.ps.uni-sb.de/pub/papers/ProgrammingSysLab/COMPOzE96.ps.gz
8- Henz, Martin. *COMPOzE Intention-based Music Composition through Constraint Programming*. http://www.comp.nus.edu.sg/~henz/talks/tai96/

# Appendix 2

# Music Generation Panel: A critical review

MOSART Workshop on Current Research Directions in Computer Music, Barcelona, Nov 15-16-17, 2001, Institut Universitari de l'Audiovisual, Universitat Pompeu Fabra.
http://www.iua.upf.es/mtg/mosart/panels/music-generation.pdf

# WORKSHOP ON CURRENT RESEARCH DIRECTIONS IN COMPUTER MUSIC

**Barcelona, Nov 15-16-17, 2001**
**Audiovisual Institute, Pompeu Fabra University**
Revised: August 2003

## MUSIC GENERATION PANEL
## (A critical review)

*Rubén Hinojosa Chapel*
Music Technology Group
Pompeu Fabra University
Ocata 1, 08003 Barcelona, Spain
+34 93 542 2104
ruben.hinojosa@iua.upf.es

Eaglestone's presentation

## Abstract

In the frame of the Workshop on Current Research Directions in Computer Music, a **Music Generation Panel** took place. The chair was **Henkjan de Honing** (NICI-University of Nijmegen, The Netherlands), who introduced and conducted the panel, whose members were (in order of appearance): **Barry Eaglestone** (University of Sheffield, United Kingdom), **Roger B. Dannenberg** (Carnegie Mellon University, Pittsburgh, USA), **Eduard Resina** (IUA-Pompeu Fabra University in Barcelona, Spain) and **Jens Arnspang** (DIKU-University of Copenhagen, Denmark). Each of the panel members made a short intervention and, after Arnsprang's words, some questions came from the audience. The panellist's answers were focused on what they have talked about.

To make available the main topics discussed in the Music Generation Panel, to the community of computer science researchers, composers, musicians, students and everybody who is interested in Computer Aided Composition, we have made a summary and have added a critical review at the end. As an introduction to the purposes of this panel, here is the

## 1. Call to the Music Generation Panel

The abundance of music generation tools and systems is well documented. These range from AI-based systems for autonomous generation of musical ideas to conventional design tools, for example, for designing and rendering of sounds. However, emerging de facto standards have been short lived, generating frustration rather than satisfaction. This panel will focus on why this is so, i.e., the extent to which accumulated results of this effort fail to satisfy the aspirations of composers. Three specific aspects of music generation will be considered. These are:

1 - Representation and contents of the product, i.e., the composition;

2 - The nature of and support for the process, i.e., creativity and composition; roles of artificial intelligence;

3 - Representation and application of individual and community know-how, including the use of repositories and archives to accumulate a history of compositional techniques used, and the use of the Web to provide open access to community knowledge.

Each aspect will be considered from philosophical, conceptual and technological perspectives. The aim is to identify open questions and unsatisfied requirements that technology has the potential to address. Many of these are partly evident in ongoing research in this area. The outcome will form the basis of a proposed scientific agenda for future composition systems research.

## 2. The Panel

A brief overview of the words by each of the panel members follows:

### -Dr. Barry Eaglestone (University of Sheffield, United Kingdom)

*(Note: Dr. Eaglestone made his speech based on his paper "Composition Systems Requirements for Creativity: What Research methodology?", so we have kept some paragraphs from that paper, and some words from his speech.)*

Electroacoustic music composition tools and systems selectively attempt to provide composers with services they require for music generation, e.g., for accessing, generating, organising and manipulating audio (and other) objects, which constitute the composition. However, a primary aim of composition software also is to create conditions in which composers can be creative in the use of these services. We believe there to be a need to establish a research base for enhancement of support within composition software for creativity.

Research into digital signal processing and the artist's use of sounds is on-going, and consequently, services relating to musical artifacts are volatile and evolving as new techniques and paradigms are integrated into composition software.

The software environment within which those services are used creatively has largely been under researched. Instead, developments have followed those of software technology. Consequently, there has been a move from asynchronous to synchronous systems, and from text to graphical user interfaces.

We believe there to be an inherent tension between principles of conventional software engineering and the requirements of creative composers. This tension can be explained in terms of models of creativity, which is often characterized by the notion of "divergent" as opposed to "convergent" thinking; the later being associated with relatively predictable logical activity and outcomes, the former with less logical and predictable activity and outcomes.

One of the most talked about and most researched area is the one which involves services for creating and manipulating musical artifacts, and there is a lot to say about that. However, we will be looking at the largely neglected area, which is the environment within which those services can be used creatively?

In our research towards this end, we are analysing data collected by observing composers at work in naturalistic settings, using methodologies ranging from software engineering through to the social sciences. What comes out is the tension between the composer's requirements and the conventional wisdom of software engineering.

Specifically, there appears to be a need for an un-typed workspace within which composition artifacts can be freely associated; support which enables composers to control the whole process, employing programming skills at the lowest DSP levels; support for interfaces which challenge the composers' conceptions, rather than reflect

them; facilities within which randomness and accidental encounters may occur; and the facility to accumulate both a personal and community repository of know-how.

The future: seeking a definitive composition system is a waste of time. It only generates insatisfaction. Software developers need to understand composers better.

I suggest two worldwide projects to the community:

1- To develop a research base for better environments that support creativity.
2- To work with the community to establish some musical grid, network and know-how base.

**Software for Electroacoustic Music Composition**

COMPOSITION TOOL COMPOSITION TOOL COMPOSITION TOOL

- Physical (Computational / Data / DSP)
- Logical (Tools / Materials / Composition objects)
- Perceptual (Freely associated untyped objects)
- Community know-how (A Grid)
- Personal Know-how (Archives / Knowledge base)
- Composition Support (What? / How? / Where? / Again? / Suggest? / Random?)

(Diagram by Eaglestone)

**Jens Arnsprang**: I think this is related with education. Composers become craftsmen, following the transition into the other side of the user environments that better support creativity. The long answer when I have the chance will be: new education.

## -Dr. Roger B. Dannenberg (Carnegie Mellon University, Pittsburgh, USA)

I would like to propose that the composition has two components:

1- Methodological applications of standard practice.
2- Creative practice, which is anything but methodological.

So, what I mean by methodological application of standard practice are techniques, maybe know-how of good works, including using digital audio, synthesis hardware and software, music notation, publishing software, also organizational materials. We record and use performances gestures and knowledge, and know about simple manipulations, such as: stretching, transposing and copying.

On the other hand, creative practice is where the music really comes from. The first rule is: break all the rules. The second is: whatever you start with, you want to think about going outside the boundaries. So, almost by definition, if there is a standard practice, you have to go outside to do something creative.

Maybe one of the most common things when people talk about creativity, is to combine things in new ways, often in unanticipated ways. So, all of these things work against any kind of methodology.

In the work that we do, one thing is the very strong tendency, especially in computer science, to try to clarify standard practice and implement it, and by doing that, we ignore the creative practice side because that is in the future work. I think approaches in that way are almost useless because it just ignores the most important part. I think computer scientists have made the same mistake over and over again.

I think there are a lot of things we can do technologically to aid the creative practice. One is building interfaces at the right level. We need to build more open-ended systems; systems that can be invoked by other programs, have options such as text input and output, so when we get a creative idea to combine systems in unusual ways, we need some way to communicate. We can make systems more scriptable, so they can do things they were not originally designed to do.

There is also a need to provide functions at many levels. What I mean here is that maybe there is an application for doing some interesting kinds of synthesis. Maybe it is great, but it is an application, and maybe what I need is a plug-in, or maybe what I need is a function library, or maybe what I need is the source code.

The final point is cost of accessibility. I think this is very critical for helping people be creative. Composers and artists are people that cannot go out and buy every piece of technology. We have this revolution of personal computers and the Internet, which give people access to so much stuff, but it is limited by expensive software applications and proprietary software, you cannot get into the source code and do creative things. It would be good for this community and the whole computer industry, to think about ways to enable artists to get access to those things.

## **-Eduard Resina** (IUA-Pompeu Fabra University, Barcelona, Spain)

What concerns me is basically what makes sound become music. Sound is a natural phenomenon, music is not. Music is an idea we impose on top of sound. Basically it has to do with the perception or the ability to perceive meaningful relationships between sonic events.

From the point of view of algorithmic composition, there are different trends. For instance, starting from some sort of mathematical logic that is not intrinsically musical, and then we want to make musical, in some way. This could be the case of fractal composition. Another trend would be just expressing some standard musical knowledge, traditional knowledge like traditional counterpoint, and implementing this into some system that makes, more or less, automatic composition.

I think there is some sort of composition where you want to start from musical intuition, but you want to set your own rules, you want to define your own musical context. And then after that, you want to be able to implement this algorithmically. In this case you

don't depart neither from existing algorithm nor from existing musical context. You have to define all of them.

It is true that certain software or certain tools allow more creative things than others. When you try to define a new musical context, a new set of rules, you take for granted that this set of rules can be implemented in some way; then you want some software that allows you to define new musical contexts. One of the main problems is that it is very easy to get lost with details when you have to be defining every step, every single thing in a composition. It happens quite often that you lose the whole idea of the musical composition, which is essential for the composer. You have a global idea, and you don't want to get lost.

One of the main problems with existing software is that, sometimes, if you have to be really powerful, you have to get down to the small details, and then you really get lost about the whole thing you are trying to work out. In general, they are not very intuitive at all for traditional musicians, who basically come with knowledge or learning of many years in musical terms, in musical concepts, and software quite often does not reflects that.

It would be essential to develop software where you can really work with musical concepts. Software has to be more intuitive for musicians, and certain solutions have to be found in this direction.


## -Dr. Jens Arnsprang (DIKU-University of Copenhagen, Denmark)

*(Note: I'm sorry, but I was unable to understand every word by Dr. Arnsprang. The main idea of his speech follows.)*

I suggest new education, a new kind of education.

Art – Computer Science – Multimedia
\ | /
What kind of education?

## 3. Our Final Remarks

Finally, we would like to make some critical comments. Firstly we disagree, to some extend, with the ambiguous use of the word "creative" made by Dr. Eaglestone. Implicitly, he divides composers into two groups: "creative composers and not creative ones". We think composers are creators by definition. Every time a composer writes a composition, he makes something new, for him and maybe for the rest of the music history. He **always** creates an "object" that never existed before: his music composition. This music shares common elements with others, but **always** has "something new" that makes it different from the rest of music created before. When this "something new" is so little, is what Dr. Dannenberg calls "methodological applications of standard practice". When this "something new" is not so little, is what Dannenberg calls, in a careful way, "creative practice".

So, we would prefer to think about composers who explore new ways of music composition and lead towards new aesthetic concepts, and composers who keep themselves, more or less, in the tradition of music composition. This is, to our mind, what Eaglestone means when he talks about creative composers and, implicitly, about not creative ones.

On the other hand, and thinking in the same direction, we would like to comment the phrase: "*create conditions in which composers can be creative*", and others very much alike.

We depart from a question such as: is there any environment, composition software, tool, etc., where a composer cannot be creative? Most music composed throughout history has been written with a pen. With this single pen a lot of composers have made contributions to music development. Great music compositions have been written, and new aesthetic concepts have been explored and developed using only a single pen.

So, should we accept that there is an environment, composition software, tool, etc., where a composer cannot be creative? If we should, maybe is time to tell composers: "forget computers, you cannot be creatives with them, go back to pen and paper times".

This happen because creativity does not reside in any tool, creativity is owned only by the musician who uses that tool. We would prefer to talk about "create conditions which **stimulate** composers's creativity" and support different and open ways of handling musical objects. The limits of creativity are in the mind of the person seated in front of the computer; nevertheless, as Resina said: "*it is true that certain software or certain tools allow more creative things than others*".

"*The future: seeking a definitive composition system is a waste of time. It only generates insatisfaction*" (B. Eaglestone). If we think about a marvellous composition system with the amazing ability to create all kind of music composition, from the past, the present, and even from the future, we totally agree: that is impossible.

Throughout the music history, composers have developed techniques, most of them algorithmic procedures, to handle all the elements of music, say: melody, harmony, rhythm, timbre, articulation, form… What a composition system does is to apply these techniques, and even new or personal ones, to the material provided by the user, i.e., the

composer. The ways to handle the elements of music are so many, almost infinite so, from our point of view, it is impossible to find a definitive computer composition system.

On the other hand, as we said before, creativity does not reside in the system, but on the composer who uses that system. We could try to model every way of music creation, and implement those models into a computer system, but who assures there will not be a composer who invents a new one?

"*Software developers need to understand composers better*" (B. Eaglestone). This is a plain truth. From our point of view, the unique way of achieving that is to learn the basics of music composition, and to work as close as possible with composers. If we want to develop medical applications, we should learn the basics (and maybe not only the basics) of medicine related to the software, and work with doctors. If we want to develop applications for astronomy research, we should learn the basics (and maybe not only the basics) of astronomy, and work with astronomers. And of course, if we want to develop music composition applications, we should learn the basics (and maybe not only the basics) of music composition, and work with composers. That is all. To understand composers better, we should think the same way they do.

## 4. Conclusions

The aim of the Music Generation Panel was to identify open questions and unsatisfied requirements that technology has the potential to address. We think this aim was achieved to some extend. Panellists clarified some theoretical aspects and proposed some directions for future research. We would like to extract what, to our mind, were the main requirements that music technology has the potential to address, and what should form the basis of a proposed scientific agenda for future composition systems research:

**Barry Eaglestone:** *We believe there to be a need to establish a research base for enhancement of support within composition software for creativity.* (As we have remarked before, "to create conditions which **stimulate** composers's creativity and support different and open ways of handling musical objects".)

**Roger B. Dannenberg:** *I think there are a lot of things we can do technologically to aid the creative practice. One is building interfaces at the right level. We need to build more open-ended systems; systems that can be invoked by other programs, have options such as text input and output, so when we get to creative idea combining systems in unusually ways, we need some way to communicate. We can make systems more scriptables, so they can do things they were not originally designed to do.*

**Eduard Resina:** *It would be essential to develop software where you can really work with musical concepts. Software has to be more intuitive for musicians, and certain solutions have to be found in this direction.*

We hope our work would be useful. Please, feel free to send any feedback, they are welcome.

# Appendix 3

# Some Projects and Reflections on Algorithmic Music

*Proceedings of Computer Music Modeling and Retrieval 2003*, Montpellier, France. Reprinted in Uffe Kock Wiil (ed.), Springer Verlag - Lecture Notes in Computer Science (LNCS) Series Vol. 2771.

# Some Projects and Reflections on Algorithmic Music

Rubén Hinojosa Chapel

Music Technology Group
Universitat Pompeu Fabra
Ocata 1, 08003 Barcelona, Spain
+34 93 542 2104
ruben.hinojosa@iua.upf.es

**Abstract.** Algorithmic Music Composition with computers (in real and non-real time) has had many approaches. It can be viewed from different points of view: scientific, technological, artistic or philosophical. In this paper the author introduces three projects developed between 1990 and 2000 in Havana, during his time as a professor, researcher and composer at the University of Arts of Cuba. He also exposes some philosophical reflections on algorithmic music systems, derived from his work at Havana and his current research in the Music Technology Group of the Universitat Pompeu Fabra.

## Keywords

Algorithmic Composition, Real-Time Composition, Interactive Music Systems, Virtual Instruments, Computer Aided Composition, Algorithmic Music.

## 1 Introduction

In 1989 Cuban composer Carlos Fariñas (1934-2002) founded, with some colleagues, the *Estudio de Música Electroacústica por Computadora* (Studio for Electroacoustic and Computer Music) of the *Facultad de Música*, at the *Instituto Superior de Arte* (University of Arts) in Havana. One year later I started collaborating with the Studio as an undergraduate student of Computer Science.

From the beginning I was assigned a project on Algorithmic Music. Our knowledge on the field was almost null, so we started from scratch, provided only with a couple of articles, some historical and anecdotal references (Hiller, Xenakis...) and a great enthusiasm in doing our best. For one decade I have been working on Algorithmic Music composition and software development and experimentation. Now I have some experiences and reflections that would like to share with other people interested in the field.

## 2 Musical Fractals (1990-1994)

Our first idea was to create a system for generating musical structures automatically, a system without a direct connection with any known musical style. Nevertheless, as we will see later, we implemented melodic transformations from traditional counterpoint, and even new ones.

Fractal images were very popular at that moment; the musical experience based on fractals carried out by American composer Charles Dodge [1] was a starting point for our research. Dodge suggested a musical structure based on a metaphorical interpretation of the self-similarity concept. He departed from the Koch curve for building parallel voices that contain proportional relationships between them, similar to those existing between the triangles of the Koch curve.

We elaborated and implemented an algorithm based on the Dodge's interpretation of self-similarity. Our first system, *Musical Fractals*, needs as a seed data, a melody, a list of melody transformations, and some numerical values such as: number of voices, values that will affect the relationships between them, etc. It computes the "piece" in non-real time, generating up to four parallel voices. One of the voices is the original melody and its variations. We implemented some interesting features that proved, through practical experiences, its strength and weaknesses. Some of these features are [2]:

1- **Scales**. The program is able to use up to fifteen different musical scales, even a user defined one, for computing the whole "piece".
2- **Traditional melodic variations.** Melodic transformations from classical counterpoint are algorithmic procedures used along centuries of musical tradition. They are powerful tools for developing a melody, so we decided to test their potential inside a computer program.
3- **Non-traditional melodic variations.** Some non-traditional melodic variations were implemented, following very personal approaches. They are:

   **Addition:** Randomly adds some notes to the melody without affecting the total length.

   **Subtraction:** Randomly deletes some notes from the melody without affecting the total length.

   **Reverse time:** It is like traditional "reverse", but it only reverses the note durations of the melody.

   **Reverse pitch:** It is like traditional "reverse", but it only reverses the note pitches of the melody.

   **Generation:** Uses a 1/f fractal noise generator for changing each pitch of the melody.

   **Simulation:** Uses a particular approach, based on *Markov Chains,* for changing each pitch of the melody. The resulting melody sounds a little bit like the original one.

   **Arpeggio:** Changes the notes whose duration is greater than or equal to a quarter-note, by an arpeggio of four notes, without affecting the total length of the melody. The algorithm uses interval values provided by the user.

   **Logarithmic:** Changes every pitch by a new one, computed with a personal algorithm that uses the logarithmic function, and involves some existing pitches.

An interesting feature that proved good results is the possibility of applying not only isolated melodic variations to the melody, but a set of joined variations that conform a much complex transformation. For instance, think about applying the following variations in order: augmentation, arpeggio, simulation and diminution. The resulting melody is only one, not four. Of course, it is possible to obtain four different melodies too! A simple command interpreter was implemented for entering coding of complex transformations. Composers found this possibility as a new and powerful compositional tool [2].

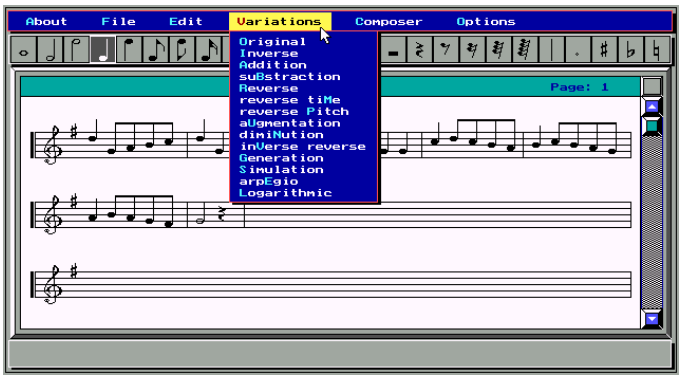



**Fig. 1.** A screenshot of Musical Fractals

A particular effort was the creation of an embedded score editor (by Claudio Daniel Ash) for entering the melody and for another upcoming projects. The resulting "piece", as well as the original melody and every transformation can be heard through a Roland MPU401 MIDI interface and an external sound module. Additionally, they can be saved to a Jim Miller's *Personal Composer* MIDI file.

*Musical Fractals* runs under MS-DOS and never was ported to MS Windows. It was awarded with some prizes. This project provided us with our first experiences in experimenting computer algorithms for music composition. One electroacoustic music piece was composed by Carlos Fariñas (*Cuarzo: Variaciones Fractales*), which was premiered in 1991 in Havana, in the frame of the Festival of Contemporary Music [2], and has been played abroad. The author made also, in 1994, a short electronic piece entitled *ET llamando a casa*, which demonstrates some of the features of the computer music system.

## 3 Orbis Musicae (1993-1996)

In 1993 we started another project with new goals in mind. We were faced with the problem of creating an interactive music system for real-time performances. The idea came about through several ways, but a very influencing one was our personal

meeting with Dr. Max Mathews in 1991, during the International Festival of Electroacoustic Music held in Varadero beach. There I had the opportunity to talk with him. Among several questions, I posed this one: How would you use a music made algorithmically? Mr. Mathews kindly answered:

"I would be interested in keeping an interaction with the algorithm; a part from the computer and a part from myself. I am interested in algorithms for improvising. With these algorithms, the musician and the computer play the music together. The algorithm chooses the notes, but the musician can select, among the options given by the program, the one he likes [3]."

With these ideas in mind we gave birth to *Orbis Musicae*, which acts like an instrument, where the musician controls different and variable parameters in real-time while the music is computed. The algorithm is quite straightforward:

There are twelve planets around the Sun moving each one on an elliptic trajectory. At the beginning each planet is assigned a grade from the chromatic scale. Then one or more triangles are placed over the orbital plane. When the planetary system starts moving, one or more planets visit the area of the triangles. As soon as a planet goes in or out from a triangle, a Note On or Note Off message is triggered, sounding on or off a MIDI controlled external sound source. The next time this planet goes inside the triangle, the original pitch assigned to it could remain the same, or could be changed to a new one, according to the initial choice of the musician. Each triangle has assigned a particular MIDI channel, so different MIDI programs (instruments) or sound modules can be controlled at the same time [4].

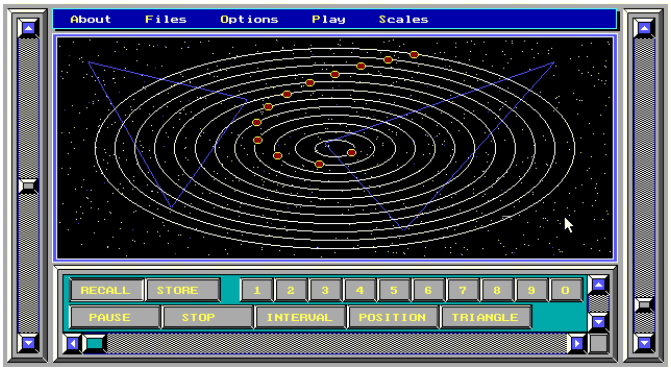



**Fig. 2.** A screenshot of Orbis Musicae

The musician can change the position and speed of the planets during the real-time performance. The configuration of the system, say: planet positions, planet speeds, assigned pitches, triangles and its assigned MIDI channels, can be saved internally at any time, and can be restored also whenever the user wants. *Orbis Musicae* uses ten memory banks, and follows the "total recall digital mixers" philosophy. The composer can use this feature for creating a scheme of configurations, which would be useful for planning the development of his piece in a sort of a score.

*Orbis Musicae* runs under MS-DOS and, as well as its predecessor, never was ported to MS Windows. An electroacoustic music piece was composed by Carlos Fariñas, who used its real-time capabilities for recording fragments of music in a sequencer. Later he took these fragments for creating a tape composition (*Orbitas Elípticas*). This music work was played in 1994 in the Bourges's International Festival of Electroacoustic Music. Another electroacoustic music work was created by Cuban composer Roberto Valera, who used the software for his real-time piece *Hic et Nunc*, performed for the first time in 1996, with my assistance, in the frame of the Havana's Festival of Contemporary Music.

*Orbis Musicae* has two essential properties:

1- It is a self-regulated system that has a personal behaviour. It can play itself endlessly without any human intervention.
2- The task of the human player is to influence the behaviour of the system, as if he were an instrument player. In fact, he is an instrument player. A player of a new kind of instrument, an active instrument. Traditional instruments always play a passive role, they react to the human gesture, but they are unable to offer the musician any musical idea by itself. At that time we used to name this kind of software active instrument, "virtual instrument".

In the middle of our investigation, we found previous experiences from other researchers whose works connect deeply to, and reinforce, the ideas we were working on. These experiences come, in one hand, from Louis and Bebe Barron, and on the other hand, from John Bischoff and Tim Perkis.

### 3.1 Louis and Bebe Barron

We found very interesting and pioneering the works done by Louis and Bebe Barron in New York, during the fifties of the past century. They intended to build new sonic models using the spontaneous electric evolution of some electronic circuits coupled between themselves, whose oscillation frequencies were placed in the audible range.

The main idea was to build series of active circuits with specific frequencies and transitory regime. By coupling these circuits to each other and influencing the behaviour of its neighbour circuits, it is possible to make changes to its own parameters. According to a partially predictable process, the union of synchronizing influences coming from its neighbour oscillators will modify the state of the oscillations of each circuit, so that they modulate their oscillations between themselves [5].

The first circuit state is dictated by external conditions, which can be changed at will. Leaving it to itself, the system of circuits follows an evolutionary process, which can be defined as the behaviour in reaction to external stimuli. This acoustic behaviour is modified according to the relationship and order established between the circuits, and confers personal characteristics to a particular considered system [5].

If we choose and study conveniently the parameters of those circuits, it could be possible to obtain an interesting sonic result, which could lead to the creation of an

electronic music composition. Under this perspective [5], Louis and Bebe Barron made music for the cinema, especially for the films *Bells of Atlantis*, *Electronic Jazz* and the science fiction film *Forbidden Planet* (1956). The soundtrack of this film is a wonderful example of artistic and avant-garde creation, and a remarkable example of the musical use of sound synthesis by modulation.

In the works by Louis and Bebe Barron we have found the notions of a self-regulated sound generation system, which owns a personal and autonomous sonic behaviour that can be controlled and changed by external influences in an interactive way. We have found also the same principle in the works by John Bischoff and Tim Perkis.

### 3.2 John Bischoff and Tim Perkis

On the CD *Artificial Horizon*, recorded between 1989 and 1990 by American composers John Bischoff and Tim Perkis, is exposed a sample of what they call "Music for New Software Instruments". In the CD booklet they express the philosophy of their music in the following terms:

"For us, composing a piece of music is building a new instrument, an instrument whose behavior makes up the performance. We act at once as performer, composer and instrument builder, in some ways working more like sculptors than traditional musicians. (...) There is another feature of the computer that attracts us: its ability to build systems of interaction with complex dynamics, systems only partially predictable, which can develop a unique "body" of their own. These woolly computer instruments can also be designed to respond to players' actions in new ways, creating a music which contains the trace of human gesture, in addition to having a degree of autonomy. In fact, for us, the distinction between composing a new piece of music and building a new instrument is not clear-cut: composing a piece of music for us IS building a new instrument, an instrument whose behavior makes up the performance. We act at once as performer, composer and instrument builder, in some ways working more like sculptors than traditional musicians. And in each case, the focus is on creating a system as open and alive as possible, bearing the precious marks of an individual character." [6]

And specifically talking about his 1978-80 piece *Audio Wave*, John Bischoff says: "AUDIO WAVE was written for pianist Rae Imamura (...). My idea was to make a live computer piece for Rae where both of her hands would be continually active, as in her conventional keyboard playing, but where her actions would serve to influence an ongoing musical output rather than have the task of initiating each sound." [7]

When we knew about the principles behind the works by the Barrons, Bischoff and Perkis, and after looking back to our experiences, we felt that we had found what path to travel through. So, we decided to build **an interactive algorithmic music system for real-time performances, not-based on any known musical style, which could act as an active instrument (self-regulated system) "where the user's actions would serve to influence an ongoing musical output rather than have the task of initiating each sound".** We were looking for a more flexible Virtual Active Instrument. Then the *Fractal Composer* project was started at the EMEC/ISA.

## 4 Fractal Composer (1996-2000)

*Fractal Composer* [8] is intended to be a virtual musical instrument for real-time performances. It plots chaotic attractors, dynamic systems and some related formulas, and makes music from these calculations while the musician introduces changes to musical parameters and listens to the results, all of this in real-time.

The system, which runs under MS Windows, features seven different fractal formulas and related algorithms for tone generation, which combine six ways of mapping pixel colors into pitches, and four note-duration or rhythms. Up to four interdependent voices may be used, conducted through three different manners or styles. Each voice owns its loudness or dynamics, its pitch limits (range) and scale. This program offers twenty-four different scales, including nine user defined ones.

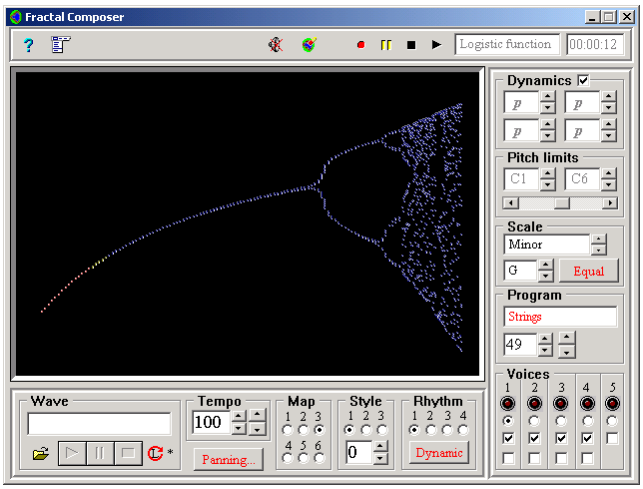



**Fig. 3.** A screenshot of Fractal Composer

The user may have control over some MIDI functions like: program changes, modulation and panning. Each voice can be moved from left to right or vice versa, automatically, at the speed the user chooses. Although program names are showed in the General MIDI convention, of course it is possible to use any non-GM external sound module. In addition to this, it is possible to load a digital sound file and play it together with the MIDI fractal music.

While *Fractal Composer* creates music in real-time, it is possible to save the resulting music, with all the changes (performance) the user has done, in a Standard MIDI File. This lets him edit his music in any sequencer or music notation software that support SMF.

The musician can store all the settings in a configuration file to be recalled later, in another session. This means that the player doesn't lose his fractal type nor its parameters, voices selected, patches, dynamics, scales, and even his own scales. A

chronometer appears in the upper right-hand corner of the display to inform the performer about the duration of his piece as time goes by.

As Xenakis said in 1971 in his book *Formalized Music*: "With the aid of electronic computers, the composer becomes a sort of pilot: pressing buttons, introducing coordinates, and supervising the controls of a cosmic vessel sailing in the space of sound, across sonic constellations and galaxies that could formerly be glimpsed only in a distant dream." [9]

The author wrote three electroacoustic music pieces with this system:

1- *El fin del caos llega quietamente*, which is intended to demonstrate that mathematics can also be a path to music. It was created entirely in real-time from the calculus of the *Logistic Function*, and recorded in one pass with no overdubbing.
2- *Satélites*. Basic sonic material was created in real-time from the calculus of the *Henon Map*. It was premiered in the XII Havana's Festival of Contemporary Music, in 1997.
3- *Oro-iña*, a real-time performance for computer, Afrocuban drum set and two dancers. It was played for the first time in 1998, in the frame of the International Festival of Electroacoustic Music held in Havana.

**Fig. 4.** Picture of the Oro-Iña performance (Photo: Archie)

## 5 Some Theoretical Reflections

### 6.1 In Search of a Satisfactory Algorithm

Algorithmic Composition researchers have tried different approaches for handling the music elements and for generating musical structures. Traditionally one of the most important elements of music has been melody. Many algorithms and models for "composing" melodies have been developed, from 1/f fractal noise to rule based or constraint programming.

Cuban composer Carlos Fariñas (1934-2002) used to say that every melody but those from monodic systems, always has an implicitly harmonic context. So, according to this idea, every random procedure for creating melodies should take this principle into account. It has no sense to look for an algorithm for creating "beautiful" or "inspired" melodies without influencing the random process by a harmonic progression.

On the other hand, when researchers intend to mimic a known music style, it is often known what elements, characteristics or procedures they should model, but what path should be follow in order to generate satisfactory musical structures not-based on any known musical style? Musicology and music composition tradition have the answer. When we were looking for a solution, composer Fernando Rodríguez (Archie) came up to us and replied: "you should try to model Analogy and Contrast".

"The notions of foreground and background (…) are critical in controlling musical flow. If similarity is in the foreground, the listener will perceive the music as continuing uninterrupted; if difference is more prominent, then the perception will be one of contrast. (…) When contrast is in the foreground, it is introduced to avoid boredom, and to deepen the listener's experience. Contrast creates emotional breadth, setting off ideas and heightening relief and definition of character. (…) Musically, when we hear familiar material in new contexts, its meaning is enriched." [10]

These reflections around melody and structure only refer to our western music tradition. It could be possible that they do not match with music traditions from other different cultures.

### 6.2 In Search of a Definitive Composition System

Throughout the music history, composers have developed techniques, most of them algorithmic procedures, to handle all the elements of music, say: melody, harmony, rhythm, timbre, articulation, form… What a composition system does is to apply these techniques, and even new or personal ones, to the material provided by the user, i.e., the composer. The ways to handle the elements of music are so many, almost infinite so, from our point of view, it is impossible to find a definitive computer composition system [11].

Every music algorithm leaves its fingerprinting in the sonic result of its execution. It has no sense to look for an universal algorithm for composing any known music style. Composers use many algorithms or algorithmic procedures everyday, and the

doors for creating new compositional procedures and new music styles are always opened, though it's no easy to travel it through. Music composition involves creativity, which is impossible to lock in a scientific model. It always flies away beyond our imagination.

### 6.3 Two Reflections About Authorship

**6.3.1** In [12] I found an interesting question that made me think: "(…) if an algorithm faithfully represents an artist's creative process, what is the difference between music produced by the artist and music produced by the algorithm?"

Algorithmic composition leads to the following situation: the user gives instructions to a computer to conceive an object (music). After a while, he receives this object from the machine. So, what now? He says: "this is my own work". Has the man stolen the object from the machine? Does this object belong to the computer?

Do not forget who has mentally conceived that object before its physical existence. Man has thought about that object, with more or less precision, before giving instructions to the machine. So, the computer has the task to give birth to the object dreamed by the man. When an artist designs a monumental sculpture, it is built by several (or even many) workers, but nobody has doubts about the authorship of the sculpture. Who is the author of the *Sagrada Familia* temple? Who denies it is Antonio Gaudí? Who denies the authorship of the *Tour Eiffel* to Gustave Eiffel?

Computers only simulate, through very strict instructions from the man, some elements of the human thinking. During the creative process, they can contribute some things to the task commanded by the man, but they only can contribute things that were thought before, things that were mentally conceived previously by the man. They cannot contribute things unconceived by the man, because they have no will nor awareness. That is the difference between music produced by the artist and music produced by the algorithm.

The man conceives and programs creation strategies, which imitate his possibilities, skills and knowledge. So, his personality will be present in the machine's results. Computers have no special artistic skills or virtuosity. They only have a representation of the skills and knowledge from the man. They are only able to mimic those human properties.

Can a machine express its individuality, its own personality, its own subjectivity? These qualities are not properties of a computer, so they cannot be expressed. Only the man can express his individuality, personality and subjectivity, from the moment he selects and gives instruction to the machine, from the very instant he conceives a music program, or when he configures the options of the software. Machines impregnate with some logic and formal characteristics the result of its computations, but the man is who gives imagination to those calculi, the man is who transmits his human sensibility with the help of a computer, and he is who transforms in art the science that could exists in automatic creations.

**6.3.2** I have found also in [12] the following interesting statement: "(…) music produced by algorithmic composition is considered somehow inferior not because it

was produced by an algorithm, but because it is someone else's music--it belongs to the *designer* of the algorithm, and not to the *user* of the algorithm."

If we accept this statement as a valid one, maybe it should be said: *Wozzek* does not belong to Alban Berg (the *user* of the twelve-tone algorithmic procedures) but to Arnold Schoenberg (the *designer*). Traditional non-computer algorithmic methods are really compositional procedures, which are always adapted by the composer to his mental scheme, to his personal point of view about music, and to his own experience and skills.

When the user configures the options of any algorithmic composition system, and gives it the seed data, he transmits his own personality, as well as when he uses any conventional algorithmic procedure, or even a rule-based music composition formalism like traditional counterpoint. So, we firmly believe that music composed with the aid of an algorithmic composition system, belongs to the user.

### 6.3 Algorithmic Music Composition: Why?

Due to the wide range of possibilities offered by computers and other electronic music devices, which are sometimes exaggerated, it is often though erroneously that usual music knowledge is unnecessary for making music with those equipments. We think computers are a powerful tool for the musician. They will help the artist in developing his ideas, in stimulating his imagination, in speeding up some technical procedures of music composition. Computers enrich the compositional process, but they will not provide the user an unexisting talent. Nevertheless, they are able to stimulate the development of an undiscovered talent or innate musical capabilities [13].

Finally, I would like to point out some general ideas related with algorithmic music composition systems I have compiled:

1- These systems stimulate the composer's creative imagination in a very new and promising way, with lots of possibilities.
2- Composition programs can handle much more data and much faster than a human composer. They let him think in a high level of abstraction, leaving low-level details to the computer.
3- They are a door for searching new aesthetic concepts, new sonic conceptions and new ways of organizing sounds. So, they are a path for music development.
4- These systems allow scientific verification of music theories, when it is intended to simulate a known musical style in order to analyse and study it.
5- They allow to better know how musical processes take place in the human mind, so they let us know better the nature of the human being.

## 7 Present and Future Work

In 2001 I was granted a scholarship from the *Universitat Pompeu Fabra* in Barcelona, for making my doctor degree in Computer Science and Digital Communication. Now I have a good opportunity to learn new things, to work on new projects and to develop the ideas I have been working on in Cuba since 1990.

In December 2002 was created inside the MTG (*Music Technology Group*) led by Dr. Xavier Serra, the IST (*Interactive Systems Team*). Having Sergi Jordà as Project Manager, the IST is integrated by: Alvaro Barbosa, Gunter Geiger, Martin Kaltenbrunner, José Lozano and myself. Now we work on a new project named *reacTable**, which puts together most of the research interests and know-how of the IST members.

*ReacTable** is a project that activates important interdisciplinary research in the field of Computer Music, which significantly departs from the MTG traditional work based on signal processing techniques. Some of the areas of research involved are algorithmic composition and real-time music creation / composition. For the near future we hope to integrate our experiences in the development of this new project.

## 8 Conclusions

Algorithmic music composition with computers (in real and non-real time) has had many approaches [9], [14–24]. It can be viewed from several points of view: scientific, technological, artistic or philosophical. We have introduced three projects developed between 1990 and 2000 at the EMEC/ISA in Havana. The first one generates musical structures in non-real time, while the other ones also generate musical structures but in real-time, in an interactive way. Neither is based on known musical styles, though they use basic musical concepts or technical procedures. To my mind, a concluding idea is the development of an interactive algorithmic music system for real-time performances, not-inspired (no mimic) on any known musical style, which could act as an active (self-regulated) instrument "where the user's actions would serve to influence an ongoing musical output rather than have the task of initiating each sound". Finally, we have exposed some theoretical reflections. I hope this paper be useful for the development of discussions and ideas related with the topics discussed here.

## 9 Acknowledgments

The author would like to thank the following people: composers Carlos Fariñas, Roberto Valera and Fernando Rodríguez (Archie), and all the students of Music Composition at the *Facultad de Música* of the *Instituto Superior de Arte* in Havana, for their contribution with ideas, suggestions, comments, discussions, critics and direct experiences with the projects I was working on during my time as researcher and professor in that place (1990-2000). Carlos A. Gonzalez Denis for having introduced me to fractal theory. Composers John Bischoff and Chris Brown for their bibliographic contribution. Composer Gabriel Brncic for his encouragement and support for making my Doctor degree at the *Universitat Pompeu Fabra* in Barcelona. Sergi Jordà for his valuable ideas and experiences as the leader of our Interactive Systems Team and the *reacTable** project. Dr. Xavier Serra for his support to my stay in the Music Technology Group and the Ph.D. program at the *Universitat Pompeu Fabra*, and for supporting the *reacTable** project.

## References


1. Dodge, Ch., Bahn, C. R.: Musical Fractals: Mathematical Formulas Can Produce Musical as well as Graphic Fractals. Byte, June 1986, p. 185-196
2. Hinojosa Chapel, Rubén: Sistemas de Ayuda a la Composición Musical. Thesis work, Faculty of Mathematics and Computer Sciences, University of Havana (1993)
3. Hinojosa Chapel, Rubén: Acerca de los Instrumentos Electrónicos, la Música Electroacústica y las Computadoras. Banco de Ideas Z / Instituto Superior de Arte, Havana (1995)
4. Hinojosa Chapel, Rubén: Sistemas de Ayuda a la Composición Musical: la Experiencia del EMEC. Proceedings of the I International Congress of Informatics on Culture, International Convention INFORMATICA '94, Havana, February 1994
5. Moles, Abraham: Les Musiques Expérimentales. Editions du Cercle d'Art Contemporain (1960)
6. Bischoff, J., Perkis, T.: Artificial Horizon. CD booklet, Artifact Recordings (1990)
7. Bischoff, John: Software As Sculpture: Creating Music From the Ground Up. Leonardo Music Journal, Vol. 1 No. 1, 1991
8. Hinojosa Chapel, Rubén: Fractal Composer: un Instrumento del Siglo XXI. Proceedings of the IV International Congress of Informatics on Culture, International Convention INFORMATICA 2000, Havana, May 2000
9. Xenakis, Iannis: Formalized Music: Thought and Mathematics in Composition. Rev. ed. Stuyvesant, New York, Pendragon Press (1992)
10. Belkin, Alan: A Practical Guide to Musical Composition
    http://www.musique.umontreal.ca/personnel/Belkin/bk/4a.html#2)%20Contrast
11. Hinojosa Chapel, Rubén: Music Generation Panel (A critical review). Workshop on Current Research Directions in Computer Music, Barcelona, Nov 15-16-17, 2001, Institut Universitari de l'Audiovisual, Universitat Pompeu Fabra
    http://www.iua.upf.es/mtg/mosart/panels/music-generation.pdf
12. Jacob, Bruce L.: Algorithmic Composition as a Model of Creativity. Organised Sound, volume 1, number 3, December 1996
    http://www.ee.umd.edu/~blj/algorithmic_composition/algorithmicmodel.html
13. Hinojosa Chapel, Rubén: Entre Corcheas y Electrones. Por Esto!, Mérida, Yucatán, Mexico, July 31, 1996
14. Alpern, Adam: Techniques for Algorithmic Composition of Music
    http://hamp.hampshire.edu/~adaF92/algocomp/algocomp95.html
15. Brün, Herbert: From Musical Ideas to Computers and Back. In: Harry B. Lincoln (Ed.), The Computer and Music Ithaca, Cornell University Press (1970)
16. Cope, David: Computers and Musical Style. Madison, WI: A-R Editions (1991)
17. Hiller, L., L. Isaacson: Experimental Music. McGraw-Hill Book Company, Inc., New York (1959)
18. Koenig, G. M.: Project One. Electronic Music Report 2. Uthrecht: Institute of Sonology, 1970. Reprinted 1977, Amstermdam: Swets and Zeitlinger
19. Kunze, Tobias: Algorithmic Composition Bibliography
    http://ccrma-www.stanford.edu/~tkunze/res/algobib.html
20. Maurer IV, John A.: A Brief History of Algorithmic Composition
    http://ccrma-www.stanford.edu/~blackrse/algorithm.html
21. Papadopoulo, G., Wiggins, G.: AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects.
    http://www.soi.city.ac.uk/~geraint/papers/AISB99b.pdf
22. Roads, Curtis: The Computer Music Tutorial. The MIT Press, Second printing (1996)
23. Zicarelli, David.: M and Jam Factory. Computer Music Journal 11(4), 1987, p. 13
24. More references: http://www.flexatone.com/algoNet/index.html

# Appendix 4

# Music compositions created with my algorithmic music systems

## 4.1 Musical Fractals

1- *Cuarzo: Variaciones Fractales*. Tape electroacoustic music. Composer: Carlos Fariñas. Premiere: Havana's VII Contemporary Music Festival, October 1991.
2- *ET llamando a casa*. Tape electroacoustic music. Composer: Rubén Hinojosa. Premiere: Havana's First Latin American Convention on Science Fiction, December 1994.

## 4.2 Orbis Musicae

1- *Orbitas Elípticas*. Tape electroacoustic music. Composer: Carlos Fariñas. Premiere: VI International Electroacoustic Music Festival "Spring in Varadero", May 1993.
2- *Hic et Nunc*. Live electronics. Composer: Roberto Valera. Premiere: Havana's XI Contemporary Music Festival, October 1996.

## 4.3 Piano Fractal

1- *Piano Fractal*. Tape electroacoustic music. Composer: Rubén Hinojosa. Premiere: Havana's XI Contemporary Music Festival, 1996.

## 4.4 Fractal Composer

1- *Satélites*. Tape electroacoustic music. Composer: Rubén Hinojosa. Premiere: Havana's XII Contemporary Music Festival, 1997.
2- *Oro-iña.* Live electronics and Afrocuban percussion. Composer: Rubén Hinojosa. Premiere: VII International Electroacoustic Music Festival "Spring in Havana", March 1998.
3- *El fin del caos llega quietamente* (1998). Tape electroacoustic music. Composer: Rubén Hinojosa.
4- *Fantasía* (2003). Tape electroacoustic music and piano. Composer: Andrés Lewin Richter.