Music Knowledge Analysis: Towards an Efficient Representation for Composition

Jesus L. Alvaro[1,2], Eduardo R. Miranda[2], and Beatriz Barros[1]

[1] Departamento de Lenguajes y Sistemas Informáticos, UNED,
JesusLAlvaro@gmail.com
bbarros@lsi.uned.es
[2] Interdisciplinary Centre for Computer Music Research, University of Plymouth, UK
eduardo.miranda@plymouth.ac.uk

Abstract. This document presents an analysis of Music Knowledge as a first step towards music representation for composition. After an introductory review of music computing evolution, several approaches to music knowledge are described: the system levels context, music theory and disciplines, dimensions in music, and finally the creative process. Then, the composition knowledge is analyzed at the symbolic level, dissecting its sub-level structure, and concluding with some requirements for an efficient representation. EV meta-model is presented as a multilevel representation tool for event based systems as music. Its structure and unique features are described within the analyzed level context. Three musical application examples of EV modeling are shown in the field of sound synthesis and music composition. These examples test representation, extension and development features.

1 Computer Music

The use of computers in music composition was introduced fifty years ago. Since then, computers have played an active role in several aspects of music creation from complex sound synthesis to automatic generation of musical material. Computer music systems, in this period, have been greatly influenced by the paradigms in which they have been developed, conditioning their creative capabilities.

As a first paradigm, the traditional conception of the *score*, as a representation for performance, has influenced several systems. They have been supported by the *score-orchestra metaphor*. Systems like *Music V* and its successors *Csound* [7], *CLM* [4], clearly reveal such dissociation by separately defining score and orchestra. The score approach lacks some representativity in two aspects. Composition elements are absent, and the sonic final result depends excessively on the specific performer.

Many of these same systems, and some others like *PD* [3] were also influenced by the *UG paradigm,* or the architecture of the first analog synthesizers. The approach is really efficient in the sound synthesis field and real time performance but not so efficient representing music for human performance. Its signal processing approach imposes a rigid lineal time conception without allowing flexible multi-temporal structures.

Some systems like *Symbolic Composer* [5] and *CM* [6] have been based on the MIDI representation. The MIDI specification was a great advance for computer music in the eighties, because it simplifies the score as a piano keystroke sequence. This simplification reduces the computation requirements, but, on the other hand, it lacks some performance information such as articulations and dynamics. It is also a score type representation for an undefined orchestra.

As we have seen, the underlying Knowledge Representation will determine and limit the creative capabilities of a computer music system, so it is worth focusing on *Musical Knowledge* as the base steps of our computer music research.

2 Music Knowledge Approaches

A first step in approaching Music Knowledge could be trying to define what we understand by the term “Music”. The simple query "define:Music" at an internet searcher can show up to forty definitions for the term. Different approaches and descriptions can be found: music as the sound itself, the organization of sounds, a human activity, an art, a communication, something impossible to define... Let's analyze some of these approaches

2.1 Music System Levels

In *"The Knowledge Level"* [2], Alen Newell describes the level structure of a system. One of the interesting properties of every level, as defined in the paper, is its independence from lower levels, so it is possible to work at one level without knowing the details of those below it. Newell also introduces the knowledge level as a zone which is immediately above the symbolic level with knowledge as its medium that can be defined or represented at the symbolic level. Figure 1 represents a possible translation of that level structure to the music domain.

KNOWLEDGE LEVEL	Emotions, Evocations Equilibrium, Continuity, Unity, Symmetry Development, Contrast, Surprise, Liveliness, Dynamism, Metaphor
SYMBOLIC LEVEL	Outlines, Elements, Procedures Composer Style, Music Theory, Harmony, Notation
IMPLEMENTATION LEVEL	Sound, Scores

Fig. 1. Levels in Music

At the implementation level, we could situate the resulting product of the composition process, that is the sound itself, or the music sheet ready for the performer.

The music message content is found at the knowledge level. It comprises what is transported by the music, and what is received by the listener. In this category, we can find components from the composer which are both conscious and sub-conscious. We can find animi states, emotions, evocations and other human components. But also more objective qualities can be found like equilibrium, continuity, development, contrast, surprise, unity, symmetry, liveliness, dynamism, metaphors...

The remaining music components can be placed at the symbolic level. That is everything that can be implemented in a music score, and can represent or transport the knowledge level. Some of the components at this important level are shown in Figure 1. A deeper analysis of the symbolic level will be presented in the next section.

2.2 Music Theory

According to Music Theory, music knowledge is something different than the knowledge level. It comprises disciplines like notation, harmony, counterpoint, music form, composition techniques like development or variation, orchestration, acoustics... In the level structure above, this music knowledge should be placed still at the symbolic level. The placement of this music knowledge or "composition knowledge" is a difficult task, a matter of philosophy and aesthetics discussion. In that sense, music is sometimes recognized as "pure music", an "always abstract art", some kind of artistic expression without represented content, and without material substance. According to this, music would represent itself, or in other words, the knowledge level of a music composition is the composition itself. This can be realized when analyzing some music compositions like Webern op.27, where equilibrium and beauty of the score architecture could be enough artistic content for the piece.

2.3 Music Dimensions

From the definition of music above, it can be deduced that pitch and time are essential dimensions of music. Other involved magnitudes are timber, dynamics, tempo and articulation. Undoubtedly, the most important dimension in music is the *time*. Every element in music is arranged in time, even sound has nothing to do without time. We can consider time as the horizontal dimension along which any other dimension is distributed. It can be easily understood looking at a music score as a cartesian representation with time as its horizontal axis. Rhythm, as the most elemental form of music, consists of the organizations of events in time. The importance of time is even more notorious when we realize that there are many time scales in music. From a microscopic point of view we can see the wave-shape of every sound note. From a macro view, the global musical form of the piece is perceived. Every intermediate depth has its significance in the music form such as notes, motives, phrases, sections, movements... An interesting property of the temporal multi-scale is the similarity of structures across scales; they share some kind of fractal recurrence.

It is also important to observe, that the score deals with time in a discrete way. Time positions and durations are constrained to a grid of bars, beats and sub-beats.

Pitch is probably the second music dimension in order of importance. Vertically arranged in the score staff, it has to do with the height of the sounds, the fundamental perceived frequency. Like time magnitude, pitch is not usually considered as a linear continuum, but as a quantized form. The piano keyboard is a clear example of this, where practicable pitch values are represented by its associated key, ignoring any intermediate pitch between adjacent keys.

Dynamics is the musical name for the sound intensity dimension. It is probably the other main dimension in music. In the score is notated by symbols like "*pianissimo*" or "*mezzo forte*" and opening or closing hairpins indicating *crescendo* or *diminuendo*, respectively. Other music dimensions like articulation or tempo should be considered in a music knowledge representation. See the references section for a further study.

Upon observing music magnitudes from the level structure described above, it is important to note how the time dimension remains stable across every level. Alternatively, other dimensions like pitch, timber or dynamics appear at some level as components representing other qualities. This property also supports the preponderance of time dimension to be considered in music representation.

2.4 The Creative Process

Composition process is more complex than a mere element association. It comprises some subprocesses in the composer world such us conception, abstraction, imagination, implementation, analysis and correction. Figure 2 is a representation of an analysis model of composition processes, shown as a cyclic process of subprocesses[1]. From a knowledge representation point of view, it is important to observe that the creative process is unique. Fortunately for arts but unfortunately for AI, no composer follows the same procedures or the same scheme. It is quite difficult to draw the frontier between the generality of the creative process and the particularity of the composer's style, language and technique.

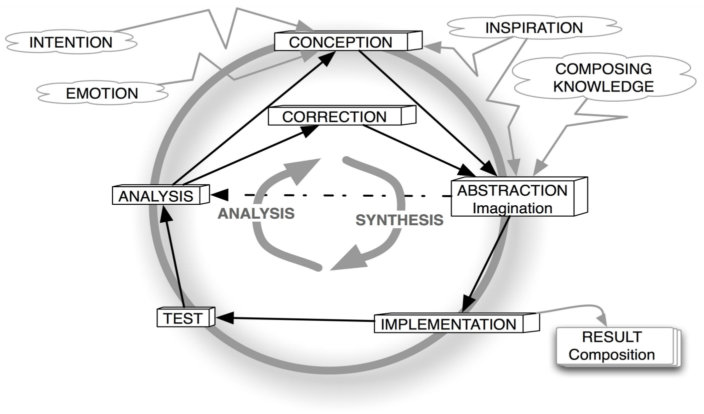

Fig. 2. Subprocesses cycle in composition

3 Composition Knowledge

Having analyzed the creation process, let's divide music knowledge into *music message* (at the knowledge level), and *composition knowledge* (at the symbolic level). It can be established that composition knowledge comprises elements, procedures, technique and strategies brought into play by the composer during the music creation process [1]. Not all elements have the same role in knowledge, but they should be ontologically organized into several layers. Elements, entities, and relationships occupy the bottom one; rules, constrains and axioms in the next one; procedures, rule-breaks, strategies, and some other meta-knowledge, occupy the next; and everything equilibrated, at the top, by criteria and intentions[1]. For now, let's explore the level structure, in this search for an efficient representation.

3.1 The Symbolic Level

The symbolic level, hosts every representation of the composing knowledge. It is the level where the whole creative process takes place. Every component, from musical notes up to piece outlines, is placed inside this heterogeneous level. Under a detailed observation of this vast area, multiple sub-levels can be differentiated, in the sense of Newell's level definition. Every sub-level keeps its independence capability, and can be described or implemented by the sub-level immediately below.

Figure 3 shows the sub-level hierarchic structure of the *Symbolic Level.* At the top, and immediately below the knowledge level, there is a "*goal"* sub-level. This zone hosts top-level decisions about the piece, like composer objectives, intentions, global musical form, time schedule, climactic points, constrains from the both chosen music language and style, and also piece conception.

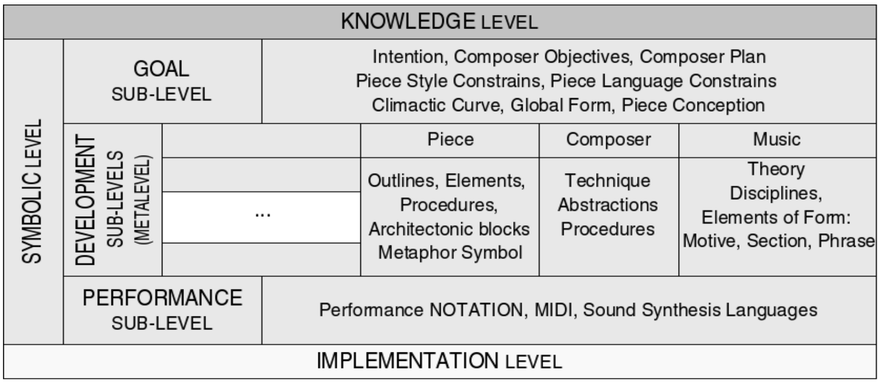

Fig. 3. Sub-levels in the Symbolic Level

At the bottom, and immediately above the implementation level, lies the "*performance"* sub-level. Performance representations, such as score notation, sound synthesis languages or even MIDI representation, can be placed in this sub-level. That is, any music representation ready to be interpreted by a performer. These representations are recognized as the final result of the process, the composition, or the piece itself.

The composition process can be seen as a trip inside the symbolic level, from the upper *goal* sub-level down to lower *performance* sub-level, passing across the region in between those extremes. This region is recognized in the *Metalevel* hypothesis [1]: "*Above performance level, there is a musical representation zone where composer usually works, and where it is possible to deal with efficient knowledge representations for computer music"*. Most of the abstraction of the composer, his technique resources, and the expertise of the music disciplines reside in the *Metalevel*. The metalevel could be also comprised of an undetermined number of sub-levels. All symbolic sub-levels share the following properties:

1. Every sub-level implies a higher degree of abstraction than the sub-level below.
2. Any representation at any sub-level can be translated into an equivalent representation at a lower sub-level. This compilation is also called *level development*.
3. Every sub-level shares the same time dimension.

3.2 Efficient Representation for Composition

At this point we are in a position for conjecturing about an efficient representation of Musical Knowledge for computer aided composition. -What features should it offer? -What design criteria should be applied? -What is the starting point? The following points of our *hypothesis for an efficient representation* for composition will try to answer these questions.

1. An efficient representation must be simple, but powerful enough to support the development of intelligent computer tools in a wide range of creativity, minimizing the limitations imposed by such representation.
2. Time dimension must be considered as the main magnitude, and must be flexibly managed at any level of depth, from micro-time to the music form.
3. It must be coherent and efficient, also at any level of representation, up from the performance level, and especially practicable at the *Metalevel*, close to *abstraction*.
4. It should contain an ontological substratum to accommodate known music entities, but flexible enough to easily incorporate new classes and relationships at higher sub-levels, when demanded by composer. Over this surface, it must be possible to represent higher layers of knowledge like constrains, procedures, strategies, in a flexible manner.

4 EV: A Multilevel Representation

EV Meta-Model [1] is proposed as a basis for an efficient representation for composition. In the described context, EV tries to model the properties shared by every sub-level in the Symbolic Level. That is, EV is a multilevel approach whose goal is the modeling capability of time dimension based representations, at any level. It is constructed upon the principles of simplicity, recursion, flexibility and coherency.

4.1 MetaModel Structure

A general overview of the EV metamodel structure is shown in Figure 4. Three main areas are represented: the ontological core, auxiliary modules and extension slots. The core was described in detail in [1]. Let's resume three interesting features of the core:

1. *Unified class approach*: every object in the system is a descendant of the same event class, so system properties can be defined in the event class definition. In addition, the event is also considered as an event container, so complex time structures can be recursively defined by single events.
2. *Liveliness character*: every parameter value in the system is, by default, a dynamic object with an evolving status. This "live" property is then transmitted to the represented music. Recursion is also present in dynamic objects in the sense that they are defined by means of simpler dynamic objects.
3. *Time relativity*: Events are provided with their own time management. That allows a time conception inside the "child-event", different from the time of its "mother-event".

Some auxiliary modules are also shown in Figure 4. The path module provides the interconnecting and referencing system. It allows any parameter of any event of the tree to be referenced by any other dynamic object. This feature expands the creative possibilities by representing relationships between any elements of the model.

The *dynamic object* module comprises the dynamic object defining system. It provides definition syntax, as well as an extendable base of predefined elementary dynamic objects.

The controlled random module provides support for the definition of dynamic objects with any random requirement. Several random distributions are provided. Every random cell status is recallable by the use of initialization seeds, so any pseudo-random behavior can be easily repeated.

Extension of the model is considered in three directions. They are represented by three slots at the top of Figure 4. Definition of new subclasses from the main event, allows customizations in the structure of events. Definition of behaviors by specializing methods extends the customizations of the level, and allows the representation of some knowledge in the procedural form. New definitions of either elementary or complex dynamic object types can be included in the base, expanding the creative possibilities.

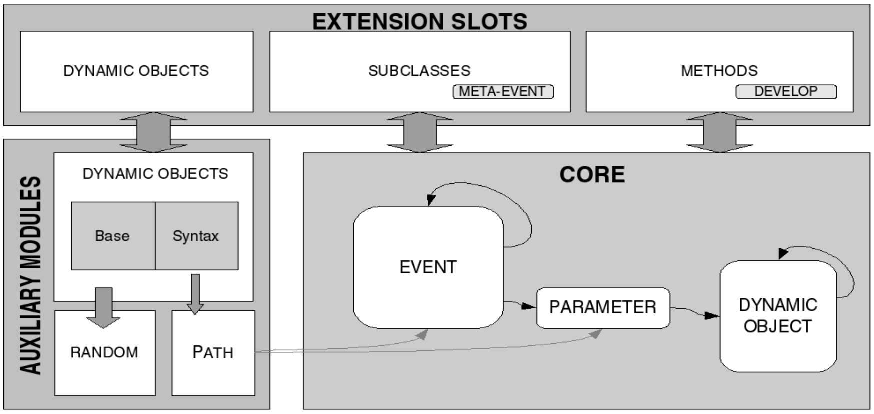

Fig. 4. EV MetaModel Structure

It is important to observe that extension can be carried out both in the metamodel or the modeled level itself. In other words, some extension could be incorporated to the metamodel, in order to be available for any other "EV model".

4.2 Level Development

As an example of a metamodel extension, the *MetaEvent* is defined as a subclass. It is the basis for the level development described below. The MetaEvent is characterized by being an event with the capability of describing new events. It operates at a higher level, and again, it allows recursion. That is, the event defined by a metaevent, could be also a *metaevent*. The level development is achieved, by providing the metaevent with the "develop" method. Developing a metaevent means creating the events described by such *metaevent*. As a consequence, a translation from a higher level to a lower level is carried out. The development could be recursive if needed until the target lower level was reached.

There are two defined subclasses of *metaevent*: the *macro*, whose expansion is done at the definition time; and the *generator*, the standard metaevent whose expansion is performed at the event time.

An interesting type of generator for music is the *"sampler-generator"* subclass. Some parameters of this type of event are descriptions of the values for the parameters of generated events in the form of dynamic objects. These "child events" are generated by time sampling those dynamic objects, and collecting the samples for each parameter into a new event. There is a direct correspondence between the parameters of both generator and generated events. This way, a large quantity of events could be described with a high degree of control by a single event in a higher level.

5 EV Modeling

Most of the work in EV modeling is done by extension, which is by defining subclasses, methods and dynamic objects. In this section several cases are briefly reviewed, as practical examples of EV modeling. Some of them are explained, with more implementation detail, in reference [1]. They are reviewed here under the analytical approach described above. Three different examples are shown, each of them probing important aspects of *EV*. They all are music applied examples. Analytically, the first example is a level development, the second shows modeling of a preexisting level, and the third is a level development of a metaevent extended model, using the model of the second example, as its development target level.

5.1 Sound Synthesis from the Metalevel

Based on the sampler-generator subclass, *EVcsound* is an application demonstrating the level expansion capability of *EV*. Its purpose is to synthesize sound from high level expressions. Csound [7], both a sound synthesis language and a compiler, is used in this application at the lowest symbolic level. Its mission is sound synthesis, that is, the compilation of sound (at the implementation level), from a procedural description of a performer (.orc file) and a note event list (at the performance sub-level). Hence the target level of the experiment is the lowest sub-level in the symbolic level.

The music composition is an event-sequence instance with a list of child events of the subclass sampler-generator. Each sampler-generator develops into a list of new child events. Development ends when all child events are of the target sub-class csound-event. The compilation to the implementation level is then performed by csound. A practical example of the process is described in [1].

5.2 Notated Score Representation

- Could traditional notation be represented by EV? **-** Could such representation keep EV flexibility? **-** Could it be extended? **-** Could it flexibly support creative higher symbolic sub-levels? These are some guide questions for the design of *Evscore*, a traditional notation compatible representation. Built by defining subclasses from the main event class, it supports the representation of musical notes, articulations, text, dynamics, slurs, voices and bars in a flexible way. Every element in the score is a descendant from the event, so they inherit all properties and features of EV. As an example, although the notated time organization of bars and beats is represented, the time flexibility of EV is still available, allowing sequencer time management, real time spotting, or even a combination of them. The extendibility is applicable in order to represent complex pitch groups like trills, mordants, glissandi or arpeggios; and even some unconventional symbols used by composer. The EV approach of this representation also implies that scores can be written from definitions of requirements at a higher level.

EVscore is more than a test. One of the key-points of this representation and also the goal of its design, is to constitute the performance sub-level for other tools, a both solid and flexible basement over which higher levels of music representation can be successfully supported.

5.3 Extending the MetaModel: Modeling a Composing Procedure

The third example is built over *EVscore*, the notation representation described above. It constitutes a test for its design goal, and it also tests(?) the metaevent extension capability of EV. *EVzone*, as the experiment is named, is a model for a composition procedure whose function is the compilation of melodies starting from the definition of some musical specifications for that melody. By using this model, the composer can try several melodies out directly from their musical specifications. In the context of the composition process model shown in Figure 2, the composer is liberated from the tedious way from abstraction to implementation, also simplifying both analysis and correction.

In *Evzone* implementation, an extension of the metamodel is carried out by defining two dynamic objects: tables and maps, and one special metaevent: the zone.

Table is an array of time sequenced value samples. Definition of a dynamic object by a set of snapshots is possible through this type of use. This is especially useful for the repeated use of either random or expensive calculation objects. In the example below, a table is used to store a random brownian shape. *Map* is used for representing recurrent structures by decomposing them into both constitutive elements and form pattern or symbolic map. This is a very musical feature based on how form is represented in the music community. The concept can be further extended by applying a grammar generator as a map, hence obtaining complex rich forms.

Zone is a metaevent subclass designed with musical form in mind. It intends to model form in a flexible manner, thus allowing multiple views of the same musical form. This is achieved by considering it not just a sequence of time zones, but a tree. The internal duration (trunk) of this metaevent is divided into zones (leaves) arranged in a tree structure, the zone-tree. The rest of the parameters are expected to be defined also as a tree. A tree correspondence is then established between every parameter tree and the zone-tree, so every definition can address the corresponding zone. The same tree structure is not needed for parameters, just any tree structure allowing a correspondence is required. As an example, an atomic parameter definition would address the whole event. In addition to extra form flexibility, zone-tree provides the capability of a unified definition of several zones at the same time, thus allowing perception a refinement of zone relationships.

```
(deftable t1 256 (br 0 1 :seed .42723))          ;brownian shape in a table
(define-zone-event my-melody                ;melody definition
    :zone-tree '((a b b c) ( a b c b c))          ;the form
    :ryth (mp zone '("+-++" "+-+-" "+---"))       ;rhythmic cells mapped into form
    :shape (mp '(a b) (list t1 (retrograde t1))) ; symmetry, map example
    :tessitura (tab t1 :min 0 :max 12)            ;step zone, scaled t1
    :ambitus (li i 5 8)                           ;melody span
    :pitchclass 'glydian                          ;the scale
    :step0pitch  'g4)
```

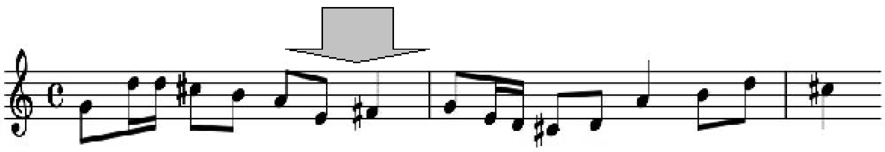

Fig. 5. Melody description example

Figure 5 shows both the musical definition, and the resultant notation of a simple melody generation example. The melody is represented by musical melodic properties like structural-form (zone), rhythm, contour (shape), tessitura, ambitus. Last two parameters, pitchclass and step0pitch, represent pitch constrains; in this case to the lydian scale rooted on G. Other constrains have been used at the implementation for optimizing scale step election. Deeper implementation details can be found in [1].

6 Conclusions

Because the creative potential of a music computer system is limited by the chosen representation, a previous deep analysis of music knowledge is a good starting point. The presented analysis is carried out in the context of system level concept. Music theory, music dimensions and the creative process, have been taken as reference points in this analysis.

Music knowledge is divided into *music message* at the knowledge level, and *composition knowledge* at the symbolic level. Composition knowledge is also comprised of several sub-levels spanning from *goal sub-level* down to *performance sub-level*. Composition process is a compilation or development of the *goal sub-level* into lower sub-levels. *Time*, the main dimension of music, remains present at every level. An efficient representation for composition should deal with time in a flexible way at any depth of scope. It must coherently support multiple levels.

EV meta-model is a representation of the multilevel nature of music composition. Constructed upon the principles of simplicity, recursion, flexibility and coherency, it sets the basis for music representation. EV modeling is performed by both extension and level development. As the given examples show, EV modeling has been found efficient in sound synthesis and music composition from high abstraction levels.

References

1. Alvaro, J.L. and Miranda, E.R. and Barros, B. (2005). *"EV: Multilevel Music Knowlegde Representation and Programming"*, in *Proceedings of SBCM, Belo Horizonte, Brazil.*
2. Newell, A. (1982). *"The Knowledge Level"*, Artificial Intelligence, vol. 18, pp. 87-127
3. Puckette, M. (1997). *"Pure Data"*, in *Proceedings of ICMC*, pp 224-227, Thessaloniki Greece
4. Schottstaedt, B. (2003). *"CLM Manual"*, Stanford, url: http://ccrma.stanford.edu/software/snd/snd/clm.html
5. Stone, P. (1997). *"Symbolic Composer"*, url: http://www.symboliccomposer.com
6. Taube, H.K. (1997). *"An Introduction to Common Music"*, in *Computer Music Journal*, Vol 21, No 1, pp 29-34
7. Vercoe, B.L. (1994). *"Csound: A Manual for the Audio-Processing System"*: MIT Media Lab