Representación del Conocimiento para la Composición Musical

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

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

Resumen. Este documento presenta el *Meta-modelo EV*, un armazón para la de representación del conocimiento musical para la composición asistida por ordenador. Comenzamos con el análisis del proceso de composición y un breve recorrido paradigmático por diferentes enfoques de la computación musical. Seguidamente apuntamos algunos elementos del conocimiento musical y perfilamos la base de una representación eficiente para la composición. Proponemos *el Meta-modelo EV* como herramienta genérica para representar estructuras basadas en el tiempo de forma coherente a distintos niveles, incluyendo altos niveles de abstracción musical. Como ejemplos de aplicación presentamos la ontología *EVScore* y la implementación de *EVcsound*, una herramienta que permite la creación de expresivas composiciones con síntesis detallada de sonido, a partir de flexibles niveles de representación superiores

1 La Composición Musical

Podemos contemplar el fenómeno musical como un proceso de comunicación, en el cual, el mensaje musical viaja desde un emisor que se corresponde al compositor, hasta un receptor asociado al oyente que percibe la música interpretada. Esta comunicación humana podría ser directa, como en el caso de la improvisación; pero lo habitual es que se produzca de forma indirecta, es decir a través de un intérprete. De este modo, podemos descomponer el hecho musical, en tres procesos fundamentales diferenciados: la composición, la interpretación y la percepción

Componer procede del término latino *"componere"* y se define como *"Juntar varias cosas para formar otra que se expresa"* [4]. En definitiva se trata de disponer una serie de elementos formando estructuras superiores, y con éstas ordenar una pieza global, o composición. Resaltemos la cualidad del *orden* en la disposición de elementos. Podemos observar el proceso de composición musical como una construcción arquitectónica de estructuras superiores a partir de elementos primarios.

La composición es un proceso algo más complejo que la simple asociación de elementos. Comprende una serie de fases y subprocesos que suceden en la abstracción del compositor, tales como la *concepción*, la *abstracción*, la *implementación*, el *análisis*, la *corrección*...

La figura 1 representa un modelo analítico del proceso de composición. En esta observación se muestra la composición como un proceso cíclico compuesto por subprocesos diferenciados. Partiendo de una intención voluntaria, o una emoción, se concibe un

elemento, una relación, una estructura, una pieza... Seguidamente se imagina, se abstrae en el contexto a partir de la experiencia y conocimiento musical, y por qué no, también a partir de la inspiración. Una vez imaginado se concretiza, se implementa y se prueba. En muchas ocasiones la implementación no será explicita, sino que bastará la mera visualización mental del resultado. En otras ocasiones se recurre al trazado de diagramas y bocetos para facilitar la prueba y el análisis. Una vez realizada la evaluación, se analiza el resultado y sus efectos sobre el resto de la composición. Como resultado, nuevas correcciones se conciben, cerrándose de este modo el ciclo.

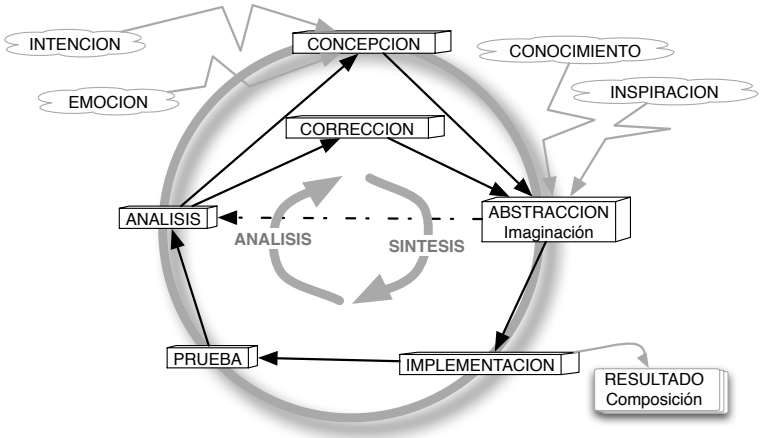

Fig. 1. Ciclo de subprocesos en la composición

2 La Computación Musical

El uso de los ordenadores en la composición musical se introdujo en los años 50. La pieza *"The Illiac Suite for String Quartet"* (1956) puede ser considerada como la primera obra musical compuesta por ordenador [3]. Desde aquel momento, los ordenadores han jugado un papel clave en diferentes aspectos de la creación musical, desde la síntesis de sonidos complejos, a la generación automática de material musical. Las soluciones aportadas como sistemas de composición musical, durante este recorrido han sido influenciadas por los distintos paradigmas de representación dentro de los cuales se han desarrollado, influencia que ha determinado, en gran medida sus posibilidades creativas

Como primer paradigma podemos resaltar la concepción tradicional de la *partitura* como medio de representación para la interpretación. En este marco, numerosos sistemas se han apoyado en la *metáfora orquesta-partitura*, es decir en la definición disociada de la partitura y del intérprete. Sistemas como *Music V* y sus sucesores *Csound* [11], *CLM* [8], SAOL [7] entre otros, manifiestan claramente esta disociación. La aproximación de la partitura supone dos inconvenientes. Por tratarse de una representación para la interpretación, deja de lado algunos elementos compositivos. Al mismo tiempo, supedita la producción final a una determinada orquesta, dependiendo en exceso de las características de la misma.

Muchos de estos mismos sistemas y algunos otros como *PD* [6] se verán influenciados también, por el *paradigma de los UG*, o la arquitectura de los primeros sintetizadores analógicos, en los cuales se configuraban sonidos, conexionando unidades elementales de proceso-generación (Unit Generator ó UG). Si bien permiten la confección de complejas

configuraciones con interesantes resultados dentro del campo de la síntesis y la interpretación en tiempo real, no se muestran tan eficaces en la representación para música interpretada por humanos. Asimismo, al estar basados en el procesado de la señal, su concepción lineal del tiempo no permite la flexibilidad de estructuras multi-temporales.

Algunos sistemas como Symbolic Composer [9], CM [10] y Bol Processor [2] han tenido como base, la representación MIDI. El desarrollo del standard MIDI, a partir de los años 80, supuso un gran avance para la computación musical, puesto que simplifica la representación de interpretación a una secuencia de pulsaciones de las teclas de un piano. Esta simplificación reduce los requerimientos en capacidad de proceso, pero impone algunas limitaciones creativas importantes al sistema que toma tal representación como base, ya que no incluye gran parte de la información de interpretación, como articulaciones y dinámicas. Asimismo se trata de una representación tipo partitura para una orquesta que queda sin definir, y que hereda las limitaciones del paradigma partitura.

La influencia de la representación del conocimiento sobre las posibilidades creativas de un sistema musical computerizado, es un factor clave que hay que tener en cuenta en las fases previas al diseño de tales sistemas. Como primer paso en nuestra búsqueda de una representación eficiente, tratemos de vislumbrar en qué consiste el *Conocimiento Musical*.

3 El Conocimiento Musical

En nuestra investigación, para identificar las piezas del conocimiento musical, partiremos de dos fuentes: el análisis de la partitura y el análisis del proceso de composición. Gran parte del conocimiento musical es representado en notación convencional, en la partitura. En el papel encontramos signos que representan elementos musicales como las notas, las duraciones y articulaciones. Sin embargo, la partitura no debe ser considerada como el producto del proceso musical, sino como un vehículo intermedio que permite comunicarse de forma eficaz al compositor con el intérprete. Será finalmente este intérprete el que contactará con el oyente. La partitura es un eficiente sistema de representación de la interpretación musical convencional, pero no una representación global del conocimiento involucrado en el proceso de composición.

Por tanto, hemos de buscar el conocimiento musical en el propio proceso de composición, en la abstracción mental del compositor, en su técnica, su experiencia, los procesos creativos formales, y por qué no también en sus intenciones. Podríamos decir que el conocimiento musical comprende los aspectos, elementos, procedimientos y estrategias que pone en juego un compositor en el proceso de la creación musical. Incluye conceptualizaciones, reglas, estrategias y transgresiones de normas; todo ello equilibrado con criterios e intenciones[1]. Analizando en conjunto el conocimiento musical, observamos cierta jerárquica en niveles, apoyados sobre la conceptualización de elementos y objetos musicales. Sobre este soporte ontológico, se apoya el resto del conocimiento musical, relaciones, reglas, y el meta-conocimiento de estrategias e intenciones Será esta base para la representación musical eficiente, el problema en cuestión que abordaremos en este documento.

3.1 Niveles de Representación en la Comunicación Musical: El Metanivel

En la figura 2 se muestran las diferentes formas de representación que el mensaje de la comunicación musical va recorriendo, partiendo desde el nivel del mundo abstracto del

compositor, y descendiendo hasta el nivel acústico que percibe el oyente. Podemos distinguir tres áreas principales de representación, cada una de ellas relacionada con cada proceso de la comunicación musical y sus respectivos agentes: *Compositor, Intérprete y Oyente*.

En el proceso, el compositor realiza internamente la traducción entre sus abstracciones mentales y el lenguaje de interpretación de la partitura. Generalmente, la compilación no se realiza de forma directa, sino que se emplean bocetos, guiones, dibujos de planificación, borradores, y cualquier otro tipo de recurso que ayude en el proceso. Es éste un largo proceso de elaboración en el que el compositor va descendiendo al nivel de notación de forma interactiva, retrocediendo continuamente a niveles superiores para realizar ajustes y conceptualizar más sus abstracciones. Es un ciclo síntesis-análisis (fig 1), una depuración realimentada e iterativa en la que es frecuente probar diferentes caminos y evaluar resultados para tomar nuevas decisiones. Desde un punto de vista de la inteligencia artificial, podríamos considerarlo un proceso de búsqueda guiada por heurísticas intenciones para satisfacer unas restricciones auto-impuestas por el lenguaje musical escogido y la narrativa planteada. En definitiva, es un largo trabajo que se realiza principalmente en distintos niveles *por encima del nivel de los lenguajes de interpretación.*

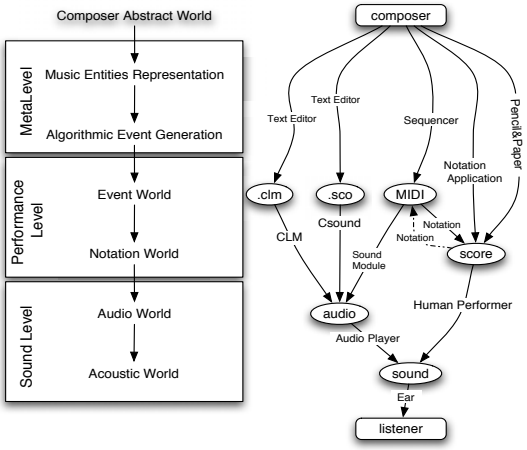

Fig. 2. Niveles de Representación en la Comunicación Musical

Establezcamos pues, la hipótesis de que existe una zona de conocimiento musical por encima del nivel de interpretación, en la que habitualmente trabaja el compositor, y en la que es posible trabajar con representaciones del conocimiento eficaces para la computación musical. Llamaremos a esta zona el *Metanivel*.

Esta hipótesis supone un punto de inflexión en la forma de afrontar nuestra búsqueda del modo eficiente en que el ordenador puede asistir el proceso de composición musical. Nuestra búsqueda de una representación eficiente para la composición musical deberá, en una primera fase, orientarse pues, hacia una representación de elementos y relaciones musicales, es decir de una ontología, flexible, capaz de tratar con nuevas entidades a distintos niveles, que permita la integración de elementos musicales y nuevas estructuras creadas por el compositor. Debe permitir la ascensión hacia niveles más próximos a la abstracción del compositor y servir de armazón para soportar herramientas informáticas válidas en múltiples niveles. De igual modo, debe considerar la dimensión de tiempo de la música en sus diferentes ámbitos, desde el micro-tiempo del sonido al macro-tiempo de la forma musical, y permitir la integración de elementos y relaciones el la dimensión espectral o vertical, sin perder la unidad y representatividad en el dominio temporal.

4 EV: Una Representación Multinivel

A continuación, presentamos el *MetaModelo EV* como sistema de representación de conocimiento en el dominio del tiempo a múltiples niveles. Nuestra propuesta supone una generalización (meta-modelo) para representar dominios en los cuales la magnitud temporal sea uno de sus ejes principales, como es el caso de la música. Uno de los puntos clave de este enfoque es su sencillez, sin dejar de ser una solución robusta.

La figura 3 representa el núcleo ontológico de la propuesta, constituido por las tres clases principales: El evento, el parámetro y el objeto dinámico. Es importante destacar la doble *recursividad* de estructuras de clases que se indica en la figura, es decir un evento constituye una serie de eventos y un objeto dinámico se construye a partir de objetos dinámicos. Esta sencillez en el diseño, no resta representatividad, sino que, al contrario, aporta una gran flexibilidad en la representación al tiempo que facilita la implementación.

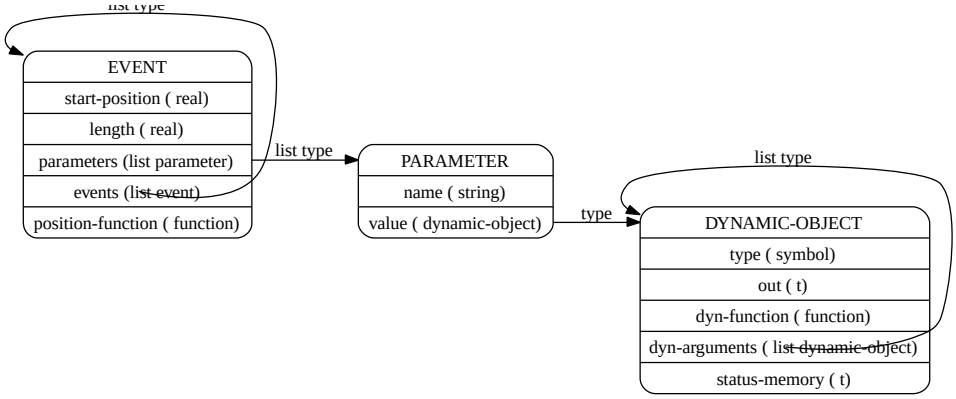

Fig. 3. Núcleo ontológico del Meta-Modelo EV

4.1 El Objeto Evento

Esta clase engloba, de forma genérica, cualquier tipo de elemento que pueda ubicarse en una posición de tiempo. Es la clase central de la ontología y como podemos apreciar en su definición, contiene cinco *slots* principales:

Start-Position es la localización en tiempo del evento.

Length representa el tiempo activo, o bien la duración del evento.

Parameters es una lista de propiedades del objeto. Estas propiedades (o parámetros del evento musical), son instancias de la clase *Parameter*. Cada parámetro lleva asociado un nombre de parámetro y un valor dinámico del tipo *Dynamic-Object*.

Events constituye el slot más importante, que hace posible la recursividad, y por tanto, proporciona la representatividad a múltiples niveles. Contiene una lista de nuevas instancias de la misma clase *Event*. Cada uno de los nuevos *"eventos-hijo"* podría contener a su vez nuevos eventos-hijo y así sucesivamente completando un rico árbol recurrente de datos a partir de un único tipo de elemento. Este concepto simplifica drásticamente la definición de estructuras temporales a distintos niveles y de relaciones entre las mismas. Puede asimismo definirse un comportamiento general mediante funciones genéricas y un comportamiento específico mediante métodos de esa función genérica para la subclase en cuestión. Se trata

pues de un armazón ontológico flexible que permite representar casi cualquier elemento musical como una instancia de la clase *evento*.

Position-Function es otra interesante característica que incrementa la flexibilidad y hace único este enfoque. Cada evento tiene la capacidad de organizar su propia magnitud temporal. De este modo tenemos un tiempo interno dentro del evento definido por su *"Position-Function"*, y una percepción temporal diferente cuando se observa el evento desde el exterior, que corresponde con la organización temporal del "evento-madre" que lo contiene. Esta flexibilidad temporal multiplica las posibilidades creativas al tiempo que permite resolver algunos problemas prácticos. Veamos 3 ejemplos:

Ej. 1: Film Scoring. Consideremos un segmento musical para una partitura de cine. La película en cuestión, podría ser vista como una instancia de la clase *evento* que emplea *SMPTE TimeCode* como organización temporal. Un determinado bloque musical de la película, es asimismo una instancia *evento* que comienza en un determinado código de tiempo y termina en otro, pero que emplea, barras de compás y pulsos métricos de la notación musical tradicional, como su organización temporal interna. A su vez, todos los eventos internos de este bloque, tales como las notas musicales, se ubican temporalmente dentro de esta organización de barras de compás. Podríamos definir *bloque-musical* como una subclase que tiene una *Position Function* o función de tiempo llamada *"tempo"* encargada de traducir aquellas magnitudes temporales. En la interpretación en vivo, el director de orquesta toma control de esta *Position Function* distribuyendo los eventos de la partitura a lo largo del tiempo real, marcando dicho *"tempo"* con la batuta. En el listado siguiente apuntamos una posible implementación mediante la definición de subclases.

```
(define-Class FILM :is-a event
  :slots ((position-function :default #'smpte2real)
          (events :multiple list :type MUSIC-CUE)
          ))
(define-Class MUSIC-CUE :is-a event
  :slots ((master-track :type master-track )
          (position-function :default #'meter2smpte)
          (events :multiple list :type MUSIC-SEGMENT)))
(define-Class MUSIC-SEGMENT :is-a event)
```

Ej. 2: Contrapunto. Podríamos definir un evento *motivo musical* y usarlo en varios momentos de la pieza a distintas velocidades, por aumentación o disminución: Una generalización del tradicional procedimiento contrapuntístico.

Ej. 3: Plasticidad Temporal. Una vez terminada la pieza, podríamos desear reorganizar la estructura temporal de la misma dilatando algunos fragmentos y comprimiendo otros: tan solo hemos de modificar su función de posiciones y definirla de la forma deseada. Como ejemplo práctico en la composición para cine, pensemos el caso en el que, una vez escrita una pieza para toda una orquesta, con el consiguiente trabajo minucioso que supone su escritura, el director de la película decide hacer un cambio de montaje importante que invalida la pieza y por tanto el trabajo realizado. Bajo nuestro enfoque, podría ser suficiente con hacer pequeños cambios en la función temporal de la instancia *music-cue*, para conservar la composición original. Incluso sería posible re-adaptar un mismo bloque musical a secuencias de distinta duración, conservando la sincronía con la imagen.

4.2 Objetos Dinámicos

El valor de cada parámetro del evento, no es estático, sino que es considerado una instancia de la clase *objeto dinámico* cuyo valor es evolutivo. ¿Por qué considerar valores constantes para parámetros que en realidad y como generalización varían con el tiempo?

Cada parámetro representa una *"magnitud"* o propiedad del evento, que se mantiene activo a lo largo de toda la vida o duración del mismo. Estamos acostumbrados a entender los datos como valores estáticos tales como números o cadenas de caracteres. Sin embargo, bajo este nuevo enfoque, los datos se suponen dinámicos en una duración, por lo que podemos considerar los parámetros del evento como *"magnitudes vivas"*.

En nuestra propuesta, se contemplan varias alternativas en la definición de la evolución del objeto dinámico a lo largo de su "vida". Para facilitar el diseño, hemos desarrollado una serie de objetos dinámicos elementales y una sintaxis intuitiva de combinación de los mismos inspirados por el lenguaje *CYBIL* [5]. A partir de objetos dinámicos sencillos es posible construir otros objetos más complejos que conservan el carácter dinámico. Entre estas unidades elementales disponibles, podemos encontrar *rectas* (*li*) entre un valor inicial -en el comienzo del evento- y un valor final -al terminar el evento- , curvas *logarítmicas* (*lo*), valores *aleatorios* en un rango (*ran, ranlo*) , *secuencias* de valores, y otros. Entre ellos, resaltemos el peculiar papel del objeto *op*, operador de *funciones*, ya que permite aplicar cualquier función definida para valores estáticos, a argumentos considerados objetos dinámicos. En el presente documento veremos un ejemplo de aplicación en la implementación EVCsound.

La sintaxis general de la definición de un objeto dinámico nos revela que la *recursividad* también se encuentra presente aquí, como forma de simplificar la implementación y de multiplicar las capacidades, puesto que cada uno de los argumentos, en la definición, es considerado a su vez un objeto dinámico. La sintaxis de definición, presenta la forma siguiente:

```
(funcion-dinámica tipo-de-dato &rest argumentos-dinámicos)
```

Para representar valores variables, podríamos haber optado por funciones en el tiempo, pero los objetos dinámicos suponen una mayor generalización. La clase *objeto-dinámico* permite albergar elementos con memoria de estado y generadores secuenciales no dependientes del tiempo, expandiendo las posibilidades hacia objetos dinámicos complejos e interesantes, tales como sistemas evolutivos y autómatas celulares.

4.3 EVScore

La figura 4 representa la ontología *EVScore* basada en nuestra propuesta. Se trata de una representación compatible con la partitura tradicional, que permite integrar elementos de niveles superiores. Observemos cómo todos las clases de la ontología son descendientes de la clase principal *evento*, por lo que heredan las propiedades descritas en la meta-ontología.

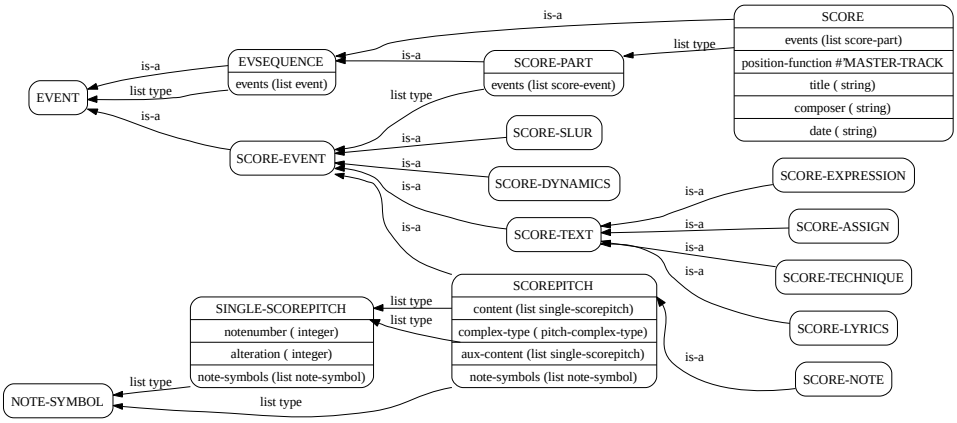

Fig. 4. Ontología EVScore

4.4 EVCsound

En esta sección presentamos *EVCsound*, una aplicación práctica basada en nuestra propuesta de meta-representación. En esta experiencia, se ha desarrollado la implementación de *EVcsound* en unas pocas decenas de líneas de código, pues prácticamente el 95% del sistema se encuentra ya disponible de forma general en el *MetaModelo EV*.

EVcsound supone un viaje desde el metanivel hasta el nivel de representación de interpretación, mediante el desarrollo recursivo de eventos generadores. Consiste en una *compilación recursiva*, a partir de un evento principal situado en un nivel superior, que se desarrolla en nuevos tipos de eventos a niveles inferiores. Estos nuevos eventos a su vez, se comportan ahora como generadores que se van desarrollando nuevamente en otros tipos de eventos de nivel inferior. El desarrollo recursivo finaliza cuando se alcanza el nivel deseado, es decir cuando todos los eventos generados son homogéneos dentro de una subclase especificada como destino. En la implementación, prácticamente todos los eventos implicados pertenecen a una subclase recursiva "*generador-de-generadores*" a la que se atribuye el comportamiento de *generador* mediante la definición de una función genérica de desarrollo, que se particulariza para sus subclases mediante *métodos*.

En *EVCsound*, el generador de partida es un *evento-composición* que se desarrolla hasta obtener eventos de la subclase de salida *evento-csound*. La secuencia de eventos sencillos de salida, constituyen la partitura *.sco* (seccion) del sistema Csound [11]. En este sistema, cada *evento-csound* del archivo *.sco* está formado por una lista de *n* valores de parámetros *p1* a *pn* asociados a un instrumento del archivo *.orc* mediante un identificador de instrumento, o primer parámetro *p1*. El parámetro p2 indica la posición temporal del evento-csound, y p3 la duración del mismo. El resto de parámetros *p4 a pn* son particulares de la síntesis de sonido del instrumento.

El tipo principal de generador empleado en *EVCsound*, lo constituye la subclase *generador-muestrador*. Se trata de un tipo de generador de eventos mediante el muestreo de categorías dinámicas, es decir, un evento con unas categorías definidas como parámetros con un valor de objeto dinámico. En el desarrollo de este evento, existe un *parámetro muestreador* responsable de determinar los valores de tiempo para tomar instantáneas del

resto de parámetros dinámicos. Con los valores numéricos instantáneos obtenidos de cada parámetro, se conforma un nuevo evento de la subclase de salida deseada.

En *EVCsound*, el parámetro muestreador empleado es el período de muestreo *period*. Este parámetro determina el tiempo que transcurre entre muestra y muestra, o dicho de otro modo, el tiempo de espera para tomar la nueva instantánea de las categorías dinámicas. En el nivel inmediatamente superior al del *evento-csound*, se han definido tantas categorías dinámicas como parámetros *p4 a pn* del instrumento, por lo que la instantánea tomada, nos proporciona directamente los valores *p4 a pn* del evento en cuestión. El siguiente código es un ejemplo muy sencillo con dos eventos del tipo *generador-muestreador*, que se desarrolla en *eventos-csound* para el instrumento referenciado por su nombre.

```
(evcsound '(0 20 superfm
                ( period (ran f .2 2)
                  amplitud-db (ran f (li f 50 70)
                                     (li f 60 80))
                  duration (op #'/  300 (pa amplitud-db))
                  pan (ran f  (lo f 0  .5) (lo f 0 .99))
                  pitch (op #'* 55 (ran i 6 16))
                  reverb-send (lo f .07  .7)
           0 25 reverb-st
                 ( reverb-time 7))))
```

El primero de los generadores, de 20 segundos de duración se desarrolla en múltiples *eventos-csound* para un instrumento llamado *"superfm"*, con un período de muestreo de valor aleatorio entre 0.2 y 2 segundos; una amplitud aleatoria en el rango que evoluciona desde 50-60 dB y 70-80 dB; un panorama aleatorio que evoluciona desde *L cerrado* hasta un rango entre *Centro* y *Derecha*; una duración entre 3 y 8 segundos que es inversamente proporcional al valor del parámetro amplitud-db; y una altura de afinación escogida aleatoriamente entre los armónicos 6 y 16 del LA1 (55 Hz).

El segundo generador, de 25 segundos de duración activa el instrumento de reverberación estéreo.

Este ejemplo representa un recorrido de un único paso, hacia niveles inferiores dentro de la jerarquía de representación, pero pueden obtenerse compilaciones desde niveles más elevados mediante la concatenación de *eventos generadores*

La herramienta *EVCsound* permite obtener detalladas partituras *.sco* que representan sonidos dinámicos y complejos, a partir de definiciones a un nivel tan alto como deseemos. De esta forma sacamos partido de la gran potencia de síntesis de csound sin necesidad de descender a la tediosa tarea de definir cada micro-evento csound. Podemos obtener piezas muy dinámicas liberando la creatividad para concentrarnos en niveles formales superiores.

5 Conclusiones

Las posibilidades creativas de un sistema musical están determinadas por la representación del conocimiento subyacente. Aunque la partitura tradicional supone una eficiente representación para la interpretación, el conocimiento musical en su conjunto, se extiende más allá de la partitura, incluyendo los elementos, relaciones, procedimientos y estrategias que pone en juego el compositor en el proceso de creación musical. Este conocimiento musical, se apoya sobre una referencia ontológica de conceptualizaciones y relaciones de las entidades musicales.

La comunicación musical se encuentra estratificada dentro de un rango de niveles desde la abstracción del compositor hasta la representación acústica percibida por el oyente. Una representación del conocimiento adecuada para la composición debe ser flexible y sencilla, pero sin dejar de proporcionar representatividad coherente a los distintos niveles. Debe representar eficientemente la abstracción musical del compositor en el metanivel, y considerar la dimensión temporal de la música en diferentes ámbitos.

El *MetaModelo EV* emerge en esta investigación como un intuitivo soporte de representación musical multinivel. Su diseño se basa en la *recursividad* de estructuras y funciones, revelándose eficaz en múltiples niveles y magnitudes. Uno de los puntos clave de su diseño es la unificación de todos los elementos musicales en un único tipo de datos, diseñado específicamente para la música. En este documento, se presentan a modo de ensayo, dos ejemplos de aplicación de nuestro modelo: *EvScore* y *EVCsound*. El primero es una ontología eficiente basada en nuestra propuesta, aplicada a la partitura y compatible con la notación tradicional. *EVCsound* es una aplicación de nuestro sistema de representación, en el que se demuestra la posibilidad de representar múltiples niveles de elementos de síntesis de sonido. Permite la composición de partituras de síntesis detallada *Csound* a partir de niveles de abstracción superiores, lo que se traduce en una gran expresividad sonora al tiempo que una flexibilidad estructural. Las piezas compuestas con *EVCsound* revelan un dinamismo y expresividad difíciles de obtener desde niveles inferiores de interpretación.

El MetaModelo EV proporciona un cimiento eficaz, coherente y flexible sobre el cual, es posible representar el conocimiento musical. Supone un importante paso adelante en el curso de nuestra investigación en la representación del conocimiento para la composición musical asistida.

Referencias

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. Bell, B. (1998). *"Migrating Musical Concepts: An Overview of the Bol Processor"*, in *Computer Music Journal*, Vol 22, No 2, pp 56-64
3. Manning, P. (1985). *"Electronic and Computer Music"*, Oxford, UK: Oxford University Press
4. Moliner, M. (1988). *"Diccionario de Uso del Español"*, Madrid: Gredos
5. Piché, J. and Burton, A. (1995). *"CYBIL Language"*, Université de Montréal, url: http://emu.music.ufl.edu/cecilia/cybil.html
6. Puckette, M. (1997). *"Pure Data"*, in *Proceedings of ICMC*, pp 224-227, Thessaloniki Greece
7. Scheirer, E.D. and Vercoe, B.L. (1999). *"SAOL: the MPEG-4 Structured Audio Orchestra Language"*, in *Computer Music Journal*, Vol 23, No 2, pp 31—51
8. Schottstaedt, B. (2003). *"CLM Manual"*, Stanford, url: http://ccrma.stanford.edu/software/snd/snd/clm.html
9. Stone, P. (1997). *"Symbolic Composer"*, url: http://www.symboliccomposer.com
10. Taube, H.K. (1997). *"An Introduction to Common Music"*, in *Computer Music Journal*, Vol 21, No 1, pp 29-34
11. Vercoe, B.L. (1994). *"Csound: A Manual for the Audio-Processing System"*: MIT Media Lab