SDIF sound description data representation and manipulation in computer assisted composition

Jean Bresson, Carlos Agon

SDIF sound description data representation and manipulation in computer assisted composition

Jean Bresson, Carlos Agon

Musical Representation Team, IRCAM – Centre G. Pompidou

{bresson,agon}@ircam.fr

Abstract

This article presents the recent work we are carrying out regarding SDIF file manipulation and SDIF sound description data representation in the Computer Assisted Composition environment OpenMusic. It explains how we use it to integrate sound processing and descriptions in the compositionnal field. Particularly, it introduces a new OpenGL based 3-dimentional SDIF file editor which we created.

1 Introduction

1.1 Sound descriptions

With all the existing sound analysis and synthesis technologies on hand, a sound can be represented in many ways. Manipulating a sound does not only imply considering it as a waveform, since it can be described by various forms of analysis or other characteristics.

These various descriptions are as many points of view from which the same sound can be considered. Each one of these descriptions will be more or less complete, will contain more or less information, will be more or less representative of the sound. If, for example, a fine spectral analysis represents a sound in an exact way, without loss of information relative to the original sound, an additive analysis may represent a less precise description but sometimes more adapted to the user's needs.

But still, there are many other existing description types for a sound sample. Some of its parameters, like evolution of the fundamental frequency, of the energy, the voiced/non-voiced character, when dealing with a sound of voice, or more complex analyses like resonance analysis, etc. can be taken into account. Thus, the representations that one can have of a sound are multiple and may be subjective. Obviously they are sometimes complementary, but generally not equivalent, and can take lots of different forms.

1.2 Musical applications

Observing these sound representations can resemble sound exploration, where one can get views of its hidden facets. This inspection, made by a composer, can be done respecting the nature of the analysis data, or in a more abstract way, in which case, starting from a given approach to the sound, from a description depending on the type of analysis he/she chose to use, a new figure or structure will appear, that may be used in a different context. However, these two approaches are not incompatible.

Figure 1 shows, in a simple example, these two different approaches. A f0 (fundamental frequency) analysis curve is used in it to give rise to two structures: a note sequence (this is the first case: frequencies are used as such since they are converted into midicents to set the notes' pitches), and a volume controller (the second case, where it's only the shape, the curve profile, which is used to construct another type of musical material).

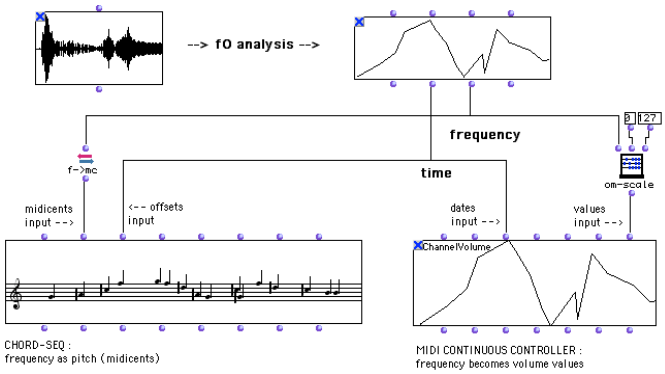

Figure 1 – Different ways to use analysis data in musical applications

Thus, in contemporary music, computer assisted composition frequently interacts with the sound analysis/synthesis domain. We can talk about links from the sub-symbolic field (signals, samples), to the symbolic field (symbolically significant entities) (Hanappe 1995). This interaction may exist by extracting and handling structures from sound analyses, as we have seen before, but also while setting the parameters of sound synthesis (Stroppa, Lemouton, and Agon 2000) (Agon, Stroppa, and Assayag 2000). Indeed, the sounds created by synthesis through computer assisted composition environments are the result of logical processes depending on parameters, and from this point of view, the structures used for the parameters of the sound synthesis can be regarded as sound descriptions, since they are what determines the sound.

Therefore, in this context, what is of an interest to us, is to approach the concept of sound description, by abstracting this description's type and origin, in our computer assisted composition environment.

2. SDIF as a generic sound description

2.1. The SDIF format

SDIF (Sound Description Interchange Format) (Schwartz, Wright, 2000) is a standard generic, open, and multi platform format for sound description data storage. An SDIF file contains one or more sequences of entities called *frames*. Each frame is dated (let *t* be the time of the frame) and identified with a type. It contains one or more *matrices*, which are the basic SDIF storage specifications. A matrix is an N * F structure, where the F columns are the various *fields* of the structure (e.g.: frequency, amplitude, phase, energy...), and N are what we call *elements* of the structure (the lines of the matrix). Thus, a *matrix* gives the values of each field for its N elements, in order to have a description of the sound at time *t*. Some matrix and frames types are predefined in the SDIF specifications, but the format is open allowing users to create their own types.

SDIF aims to unify the sound description formats. Some analysis tools, like *Diphone Studio* (Lefevre, Rodet 1997) or *SuperVP* (Depalle, Poirot 1991), already allow exporting their analysis in SDIF format.

2.2 Data abstraction in OpenMusic

Thus, we are dealing with a file format which makes possible the storage of an unspecified sound description or analysis, and proposes a strict and normalized structure. For this reason, we propose to regard SDIF files as an abstract support for sound descriptions. By this abstraction, we will try to create a new object model for the musical composition.

Using the *OpenMusic* software (Assayag et al. 1997) (Agon 1998), we can allow this model to be integrated in the compositional process. Among the other objects it is able to handle, OpenMusic has an *SdifFile* class. The purpose of this work is to integrate this object, which represents our sound description abstraction, in the composition environment as musical material. We mainly carried out this integration by equipping it with a graphic editor, but also by creating various OpenMusic tools, allowing data conversions towards SDIF format, or extracting parts of the data contained in an SDIF file.

Figure 2 shows a quick example of how can SDIF be used to store analysis data in OpenMusic. The *FFT* box calls the *SuperVP* engine to make an FFT analysis, with specified analysis parameters and on the selected area of the input sound. The resulting analysis is stored as a SDIF file specified in the output of the *FFT* box. Next, we have our SDIF box representing the stored analysis data. Further on, we will see how is it possible to visualize it using the SDIF editor. Below the SDIF box, the *getSdifData* function box extracts local data from the SDIF file and formats it as lists, to be used in OpenMusic.

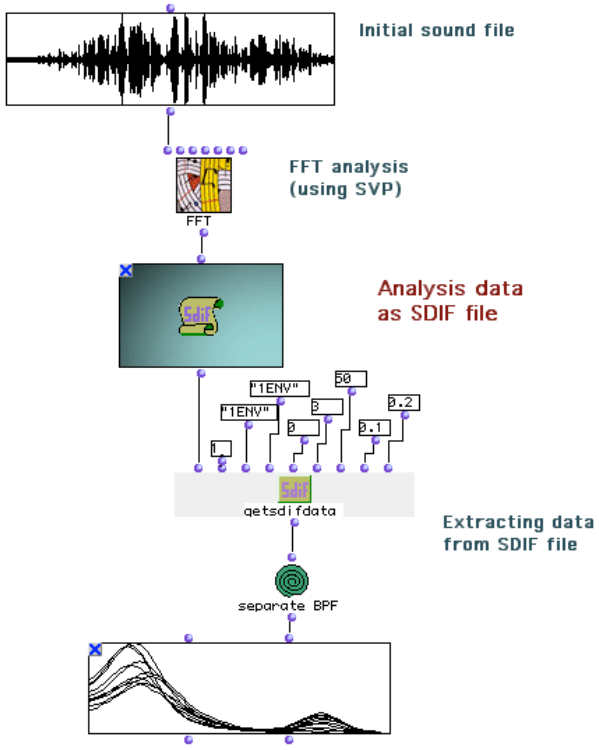

Figure 2 – FFT analysis stored as SDIF and file data extraction in OpenMusic

In this way, SDIF sound description data can be both results from a downstream computation, and also used as musical objects in the composing environment, becoming basic material for the algorithms. Thus, we do not deal with

a simple sound analysis anymore, but with a symbolic object taking an integral part in the structure of composition processes.

3 An editor for SDIF files

3.1 Visualization of SDIF data

In order to use SDIF files as relevant objects in the compositional environment, a graphical editor seems to be necessary. Indeed, accesing such an editor in OpenMusic will allow the composer to visualize and control the sound descriptions data he/she owns. The object issued from these data could be visually appreciated. Modifications it would undergo could be evaluated, and eventually be reflected on the continuation of the execution. Such an editor will also make it possible to locate significant or interesting parts in the file. In the Figure 2 example, it would allow to choose the values to set the getSdifData box inputs, which determine the data area to be extracted from the SDIF file.

Indeed, one of the characteristics of analysis data is the great quantity of information contained in it (depending on the sampling rates, and on the description's precision), which differentiates its approach from the symbolic approach of the models generally used in computer-assisted composition. This characteristic is an important aspect in handling sound description data in musical composition, since one wishes as far as possible to hold and manipulate both reduced and representative data. A part of the composer's work then consists of extracting, from this set of data, lighter structures which make sense and are easy to handle in a musical context. This data reduction, made by selection, thresholding, or filtering, requires a preliminary localization, which will be facilitated by a graphical representation.

3.2 SDIF-Edit

SDIF-Edit is the graphical editor for SDIF files which we developped. Written in C++ and completely based on OpenGL, it can be used as a standalone application, or be launched from OpenMusic as an editor for an SdifFile box, like the one we mentionned in the example of Figure 2.

This editor allows, in a first step, a high level navigation in the file infrastructure: the different sequences (flows) contained in the file, matrices inside each sequence,... To allow this, the file is firstly entirely parsed, in order to create a file descriptor, which stands for a kind of map of the SDIF file, ignoring (for the moment) the numerical data values. The user can then get information about the different element types, composition, size, and then select, in the recorded file data structure, the matrix sequence he wants to visualize (see Figure 3).

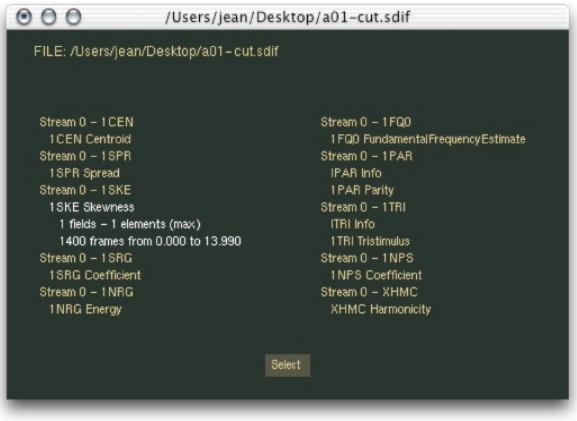

Figure 3 – SDIF Matrix selection in SDIF-Edit

Once the desired matrix flow is choosen, the proper numerical data are loaded into the application storage structure. This structure is like a 3D table. At each moment, a matrix and a time value are stored, and the matrix itself is a table whose dimension is the number of elements it contains. Finally, each element of the matrix table contains a value for the different fields of the matrix.

The next step is the visualization-edition of this numerical data we've just loaded (figure 4 shows the aspect of the viewer).

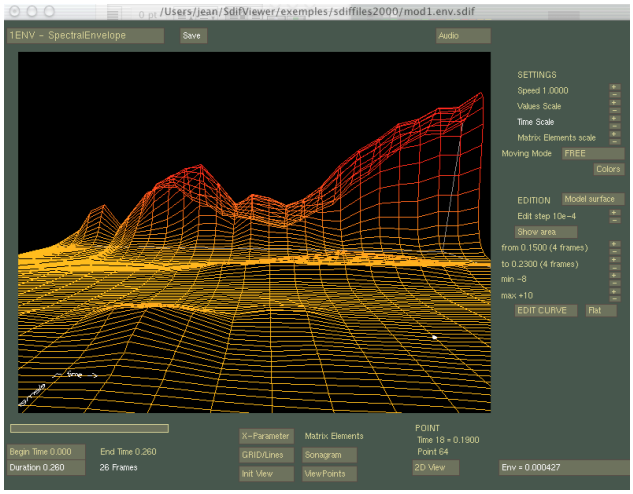

Figure 4 – SDIF-Edit: 3D SDIF file editor.

By respecting this method of data processing, we take advantage of the SDIF data abstraction, remaining free of any consideration regarding the nature of the handled data.

3.3 Data visualization and edition

A 3D space is thus created (see Figure 4), in which the data are drawn as an elevation grid. The horizontal dimensions stand for the matrix lines (called *elements*), and time. Elevation represents the numerical values of the data.

Color indicates values as well. Initially, the color of a 3D point is calculated by linear interpolation between two colors corresponding to the grid's maximal and minimal values, but we will see that this parameter is adjustable.

In this 3D space, the camera (i.e. the user) can move according to several modes of navigation (either with free movements, or by rotations and translations centered on a particular, previously selected, point of the grid). Mouse and keyboard are both used to make the movements easier, and the graphical user interface allows to set the visualization's parameters, the scales in the three dimensions, in order to focus and record points of view on zones of interest.

We carried out several edition techniques on this 3D data grid, making it possible to correct errors, and to modulate and personalize the structures' shapes. To edit the 3D grid, we must first select a point using the mouse. Then it is possible to extend a zone around this point to determine the grid area to edit. The numerical values are then modified either as a single rigid block, or by respecting the initial grid morphology (the modification of the central point will affect each one of the other points according to their initial elevation difference), which has for effect to accentuate or attenuate the existing data variations (we think for example about applications in re-synthesis). The initial edition step is calculated using the range of grid values range, but is finely adjustable.

Another special edition method has been developed, using a curve profile editor (see figure 5), which allows, by using techniques of spline curves and surfaces, to determine a 3D deformation profile starting from a set of 2D control points. The modification applied to the central point will then be reflected on the selected data zone according to this deformation profile.

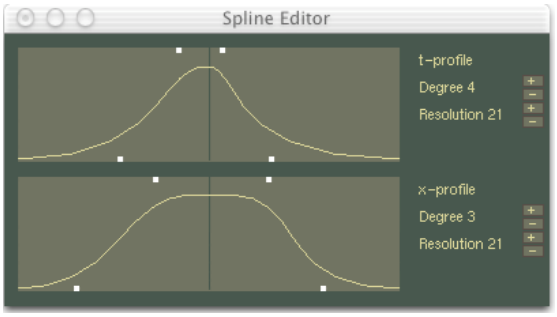

Figure 5 – A spline curves editor to determine the 3D edition profile

Starting from this free 3-dimensionnal navigation and edition system in the virtual data space, personalized and subjective images and views on concrete data are reached, opening the passage towards the symbolic world.

4. SDIF Data in a 3D space

4.1 Limitation of the representation

SDIF-edit provides a three-dimensional representation of sound description data. However, an SDIF matrix sequence would be a four-dimensional object. As mentioned before, a matrix is made of a set of columns representing the analysis fields (e.g.: phase, energy, amplitude, frequency...) So a matrix could be represented in three dimensions as a set of curves representing the various (or the single) fields. But the other dimension is time: we are generally dealing with time-sampled data.

Considering that there is generally no continuity between the various fields (the matrix columns), one can imagine that their simultaneous representation if of little interest. It makes more sense to represent the temporal evolution of a given field. Hence, we have decided to focus the representation on one description field only, with the possibility to switch to the other ones using the user interface buttons (our internal data storage structure allows this without accessing the file again). Therefore, we are making a projection of our data in a 3D space, according to one of its different fields.

4.2 Representing various matrix fields

We've seen that our 3D space represents one descriptional field (a column of the SDIF matrices), and that it's possible to switch from one to another easily. Figures 6 and 7 show an example of switching between two fields (frequencies and amplitudes) of an SDIF spectral analysis.

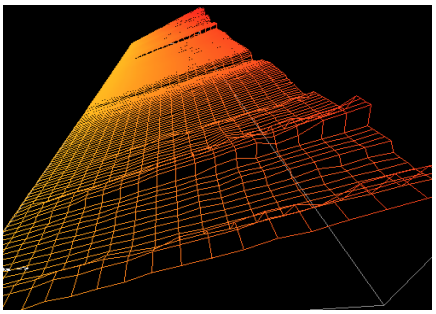

Figure 6 – Frequency field of a spectral analysis

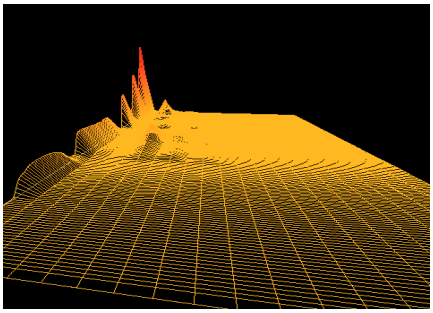

Figure 7 – Amplitude field of a spectral analysis

In some cases, a close relationship may exist between different fields (for example the case we have seen on figures 6 and 7, with frequency and amplitude), so that we may want to include them in the same representation. At each point of the data grid representing a field, the user interface provides the numerical values of the same point in the other fields. Then we go further, proposing to align transversal elements of the representation with the values of another field. This is done by positioning our 3D points in the transversal axis corresponding to the desired field values instead of the original regular spacing. This operation deforms the grid and will give the graphical representation of the first field according to the second (e.g. amplitude according to frequencies) and its time evolution.

In Figure 8, we can see the example shown before in figures 6-7, where amplitudes have been aligned to the frequency field. The representation is more significant in the sense that visually aligned points correspond to a same frequency.

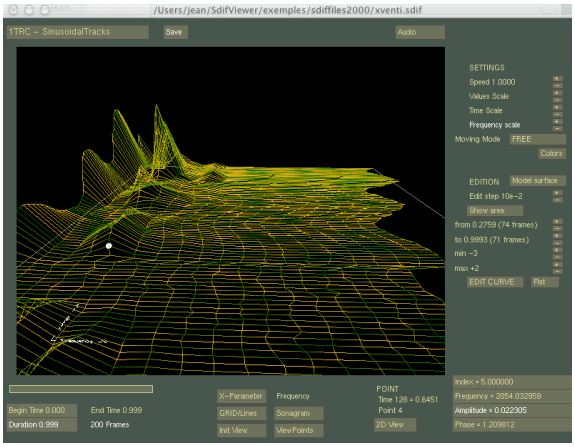

Figure 8 – Spectral amplitude representation, aligned on the frequency field and colored according to the phase.

In order to make sense out of this operation, the field on which the representation is aligned must have a monotonic evolution (which is generally the case for frequency curves, for example). If this evolution is linear and constant, the result will be an image similar to the initial one, since the transversal positions of points remain regular.

The representation is completed by the use of the color parameter, which adds information coming from another field. The color of the points will not necessarily depend on the values of the main visualized field, but possibly of another field of the matrix. Figure 8 illustrates this possibility: color depends on the phase value in each point of the graph.

4.3 Special cases

The three-dimensional representation is still interesting while dealing with non-temporal analysis, or when it concerns the evolution of a single parameter. These are frequently encountered cases, and, a priori, bi-dimensional problems. We could mention resonance analysis (*Resan*) (Loizillon 1999) as a non-temporal analysis example, and descriptions such as fundamental frequency estimation, energy, as a temporal evolution of single parameters.

In these cases, the possibility of moving and configuring the view of the graphical representation of data makes possible a visual observation of the sound, under new personalized angles (see Figure 9).

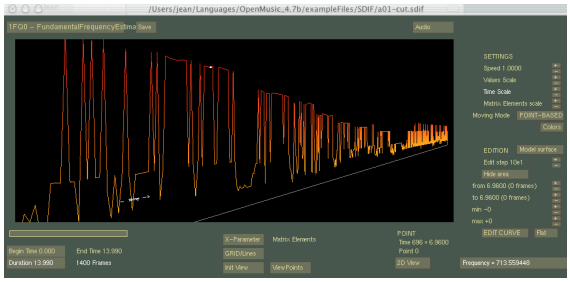

Figure 9 –SDIF-Edit with bi-dimensional analysis data

5 Alternative data representations

5.1 The duality problem in the SDIF data representation

One interesting problem in the SDIF data representation resides in the format flexibility, from which results the necessity to set aside any prior consideration or estimation concerning the content of the file to visualize (origin of the data, interpretation, size...). However, it may be important to propose visualization and edition possibilities adapted to specific needs inherited from some frequently used data types.

This duality thus forces us to conceive intuitive and effective navigational systems, taking place between the data genericity provided by the SDIF format and the necessary specificity for a sufficiently explicit and significant representation.

The general three-dimensional visualization remains valid independently of the data type used, and thus is completed with specific views, dedicated to particular data types. These points of view were conceived considering alternative data representations.

Thus, the user is moving between abstraction and specificity paradigms, determining his positioning by the choice of the representation(s) he uses.

5.2 2D projections

Data can firstly be visualized and edited as 2D curves, being transversal slices of the three dimensional data grid at a given point's coordinates. These 2D curves allow following the temporal evolution of a single parameter, or visualizing the aspect of a complete field at a given time. Figure 10 shows an example of the 2D view editor window. This window can be an overview of what could look like some data extracted from the SDIF file (see example on Figure 2).

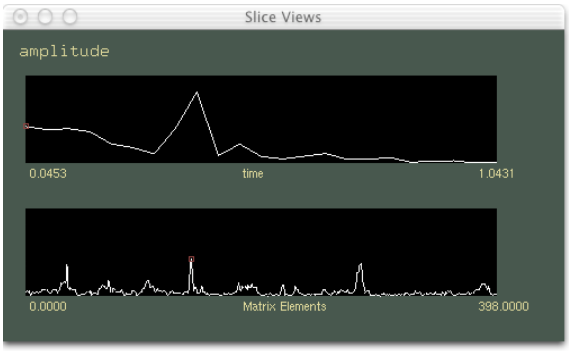

Figure 10 – 2D views of SDIF data

A sonogram editor carries out another plane projection of the data (Figure 11). This representation is particularly dedicated to spectral data. It displays a standard spectral representation of the sound analysis.

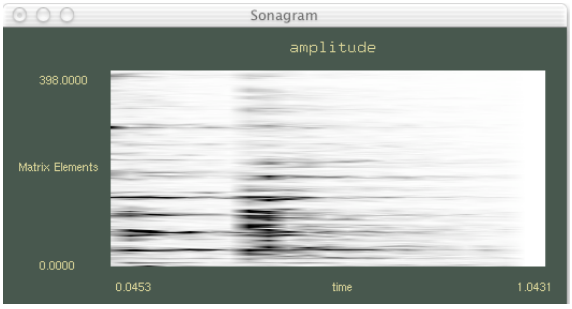

Figure 11 – Sonogram representation of SDIF spectral data

5.3 Audio representation

Another experimental representation has been included to the editor, using auditive methods.

The SDIF data first can be associated with a sound (normally, the original sound corresponding to the description). Playing this sound, or selected parts of it, can be a good reference to understand and localize data.

SDIF-Edit also includes a mini-synthesizer, able to generate and play sinusoidal signals with given frequencies and amplitudes. So the user need only assign matrix fields to these parameters, and then can listen to the 3D data variations as frequency or dynamic variations of the sinusoidal signal.

6 Applications in musical composition

6.1 From analysis to abstract material

Let's consider an example from the composer Karim Haddad:

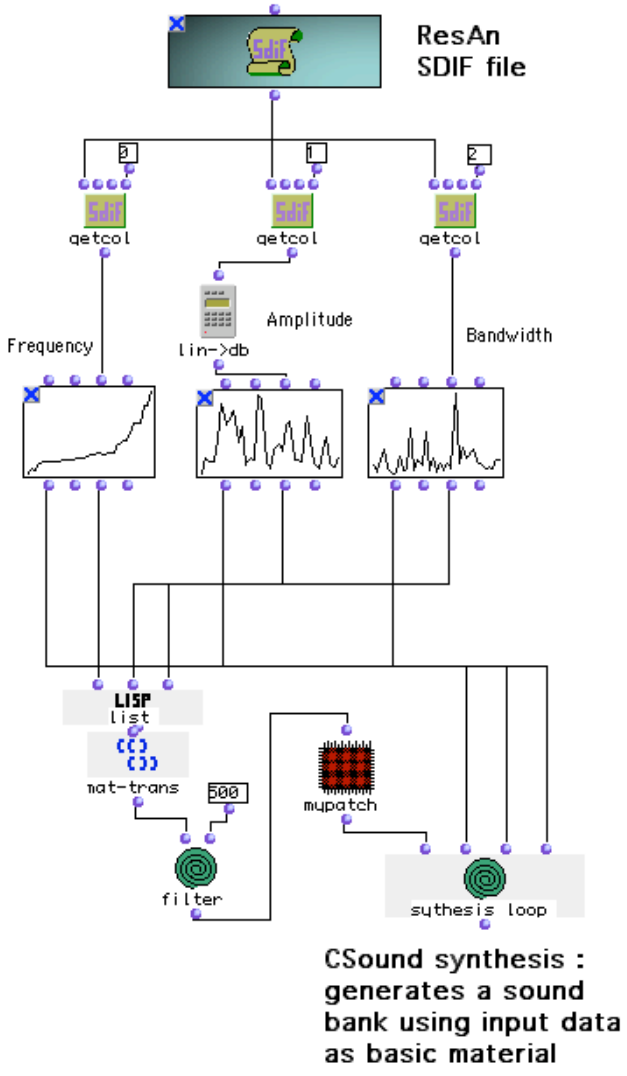

Figure 12 – Example of SDIF data used as synthesis material in OpenMusic

In this example (Figure 12), K. Haddad uses data he previously localized in his SDIF file (a Diphone-ResAn

analysis), and extracts three break-point functions from it. These functions will be used as the base material for a random generation process, producing a sound bank (we hid the biggest part of the algorithm in the "synthesis loop" box, to keep the figure easy to understand).

Various functions allow data extraction from the SDIF file, either by asking for frame and matrix types and then for time or element ranges in the matrix, or directly by asking for a frame, matrix, and column number.

We can see in this type of application that the SDIF tools in OpenMusic make a significative link between sound analysis and the symbolic compositional environment.

We could then look at Figure 13, which emphasizes the data abstraction idea showing how, using the same data from the SDIF file, K. Haddad creates a rhythm structure used to set the temporal disposition of the sounds previously synthesized.

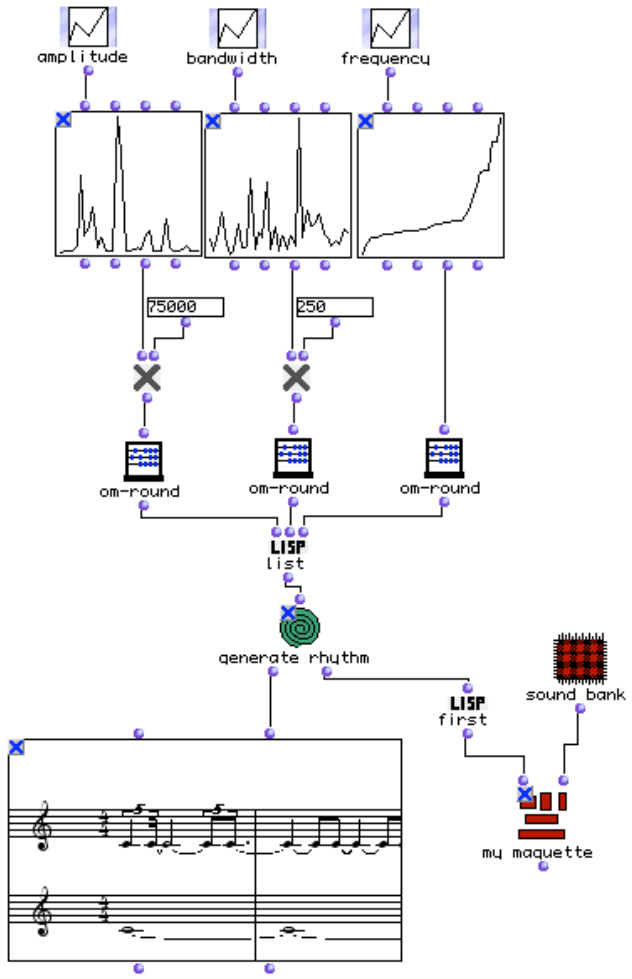

Figure 13 – Creating a rhythm starting from the same SDIF analysis data.

6.2 SDIF with partials

In musical applications, we are often interested in partial tracking analysis, which consists of extracting the main partials from the spectral analysis. Partials have a direct musical signification, since they are composed of temporal situations and frequential and intensity evolutions. This type of information fits particularly with high-level manipulations and processing in musical composition.

A special processing allows dealing with partials data in OpenMusic and SDIF-Edit. We used the SDIF format flexibility to create a new SDIF type, introducing a more symbolic conception of SDIF data, adapted for partials representation. Such data can result from AudioSculpt partials tracking (Hanappe 1995), from which is based the SDIF conversion algorithm, or from personal inputs in OpenMusic.

This special data type involved a special processing for the representation: we do not have sampled data anymore, but a set of symbolically described objects. According to the distinction we established in section 5.1, this representation positions us in the specificity paradigm.

Figure 14 is a 3D representation of a partial set in SDIF-Edit.

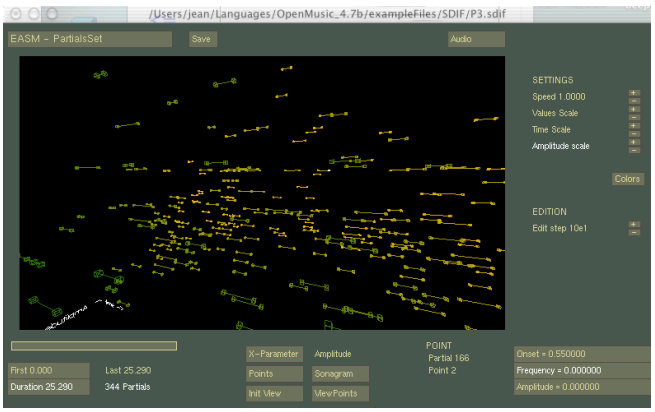

Fig 14 – Partials representation using SDIF-Edit (the vertical axis represents frequencies; the two horizontal axis are time and amplitude).

In the 3D partial space, the user has the same navigation possibilities as in a classical view. He is now "flying" between symbolical objects extracted from a sound analysis, and/or dedicated to the creation of a real sound by a synthesis processing. Data can be edited as well, either point by point, or as a whole partial (a partial can be made of more than two points).

SDIF storage and representation of partials is an interesting challenge, since it fits well with our symbolical approach of sound description data, and has fundamental interests in synthesis applications.

The Figure 15 is an example of an OpenMusic patch using partials in a SDIF file. These partials come from an AudioSculpt partial tracking. They are represented with SDIF-Edit in Figure 14. In this OpenMusic patch, the partials were enriched using OmChroma library (Stroppa, Lemouton, Agon 2000) by adding secondary partials around each one of them, and finally were reinserted in a synthesis process in order to generate a new sound, having a similar harmonic and rhythmic structure as the initial one (the original sound from which the partials were extracted), but with different perceptive characteristics.

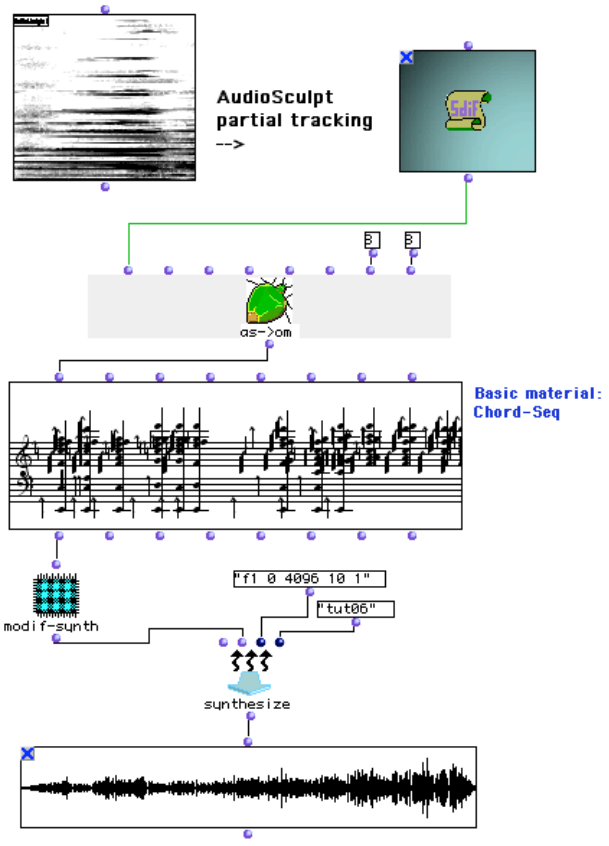

Figure 15 – Example of synthesis in OpenMusic using SDIF partials

7 Conclusion

As we have explained in this article, we are trying to make the connection between sound processing and computer assisted composition more flexible, by introducing sound analysis data and manipulation in the compositional environment, and by creating abstractions in order to communicate with sound analysis and/or synthesis applications.

We believe that a graphical control on this type of data, through the SDIF format and the SDIF-Edit application included in OpenMusic, can improve subjective appreciations and influence the composer's aesthetic choices while he/she would try to pass through the usual border between these two musical fields.

SDIF-Edit is included in the OpenMusic distribution, and is also available on the IRCAM Musical Representation Team web page at :
http://www.ircam.fr/equipes/repmus/bresson/sdifedit.html.
The application runs on MacOSX platform. Sources can be downloaded and compiled on other platforms.

8 Acknowledgments

We would like to thank Karim Haddad and Marco Stroppa for their collaboration and musical expertise on this project.

SDIF has been developped by IRCAM (Centre G. Pompidou, France) and CNMAT (Berkeley, US). Free SDIF librairies for Unix, Windows, and Macintosh can be downloaded from http://www.ircam.fr/sdif and http://www.cnmat.berkeley.edu/SDIF.

References:

Agon C., M. Stroppa, G. Assayag (2000) "High level musical control of sound synthesis in OpenMusic", *Proc. ICMC 2000, Berlin.*

Agon C. (1998) "OpenMusic : Un langage visuel pour la composition musicale assistée par ordinateur" *thèse de doctorat de l'Université Paris 6.*

Assayag G., C. Agon, J. Fineberg, P. Hanappe (1997) "An Object Oriented Visual Environment For Musical Composition" *Proceedings of the ICMC 97, Thessaloniki.*

Depalle Ph., G. Poirot, (1991) "A modular system for analysis, processing and synthesis of sound signals", *Proc ICMC, Montreal, Canada, 1991.*

Hanappe P. (1995) "Intégration des représentations temps/fréquence et des représentations musicales symboliques", *Mémoire DEA ATIAM, Ircam/Paris VI.*

Lefevre A., X. Rodet (1997) "Diphone" *Proc. JIM97, Lyon, France.*

Loizillon G. (1999) "Diphone Studio V 2.8, Obtention de Modèles de Résonnance" *Rapport interne, Ircam.*

Schwarz D., M. Wright. (2000) "Extensions and Applications of the SDIF Sound Description Interchange Format". *Proc. ICMC 2000, Berlin.*

Stroppa M., S. Lemouton, C. Agon (2000). "omChroma, vers une formalisation compositionnelle des processus de synthèse sonore" *IRCAM, Centre G. Pompidou.*