Modèles de structures tonales dans Elody

Stephane Letz, Dominique Fober, Yann Orlarey

Modèles de structures tonales dans Elody

Stéphane Letz, Dominique Fober, Yann Orlarey
GRAME
9, rue du Garet 69201 LYON,
{letz,fober,orlarey}@rd.grame.fr

***Résumé** : Elody est un environnement de composition musicale basé sur un langage de programmation visuel dérivé du lambda-calcul et sur une interface à manipulation directe. Le langage de programmation d'Elody peut être vu comme une **notation musicale agissante**, construite à partir d'un langage de description du domaine musical étendu avec les concepts d'abstraction et d'application du lambda-calcul et où les abstractions peuvent être organisées en partitions avec une **sémantique applicative**. Nous montrerons comment des connaissances musicales de plus haut niveau peuvent être définies dans le système et mises en œuvre dans des **modèles opératoires** de pièces de musiques tonales.*

1. Les langages de composition musicale

Les langages informatiques ont été utilisés dès les débuts de l'informatique musicale, dans un premier temps pour l'analyse et la synthèse sonore, puis très rapidement dans le domaine symbolique. Les outils d'aide à la composition, les techniques algorithmiques et l'utilisation de divers modèles mathématiques ont permis d'explorer de nouveaux domaines de la composition. Ils ont suscité l'apparition de nouveaux modes d'investigation de la pensée musicale, en s'affranchissant des procédés classiques de composition pour explorer une grande variété et une grande richesse de modèles de pensée.

La composition assistée par ordinateur moderne s'est développée comme une discipline à part entière en suivant les progrès de l'informatique : nouvelles interfaces graphiques, disponibilité d'ordinateurs personnels, et bien sûr grâce à l'évolution des langages de programmation eux-mêmes. Dans les nombreux environnements qui ont été développés, différents paradigmes nouveaux ont été utilisés : le modèle objet avec les travaux S.Pope, de D.Oppenheim [Oppenheim 1989] ou F.Pachet [PRC 96] avec Smalltalk, la programmation fonctionnelle [HMGW 96] [OFL 97], la programmation par contraintes [BR 98], ou l'association de paradigmes multiples [Agon 98].

Les langages informatiques ont suivi des évolutions conceptuelles importantes. On est passé de spécifications impératives où les données et les programmes sont très fortement séparés vers une vision plus déclarative où on dispose de concepts plus puissants. Par exemple la notion de *types de données abstrait* regroupe dans une même entité les structures de données et les opérations qui les manipulent. Ceci correspond à une vision plus naturelle des concepts, qui ont souvent une réalité multiple, pensés comme des données à certains moments et manipulés comme tels, et à des processus à d'autres moments. Ceci permet aussi des spécifications de plus haut niveaux ou certains détails du calcul n'ont plus besoin d'être décrits.

Dans le domaine musical, la même évolution s'est produite dans les langages de composition. On est passé de la vision *données musicales statiques manipulées par des processus externes* pour aller vers des modèles plus déclaratifs. Dannenberg décrit par exemple le langage Canon comme un *croisement entre un langage de programmation et une notation musicale, les partitions sont elles même des programmes* [Dannenberg 89]. Ici apparaît d'ailleurs la question importante de la notation. Utiliser tel ou tel paradigme de programmation pour décrire des structures musicales n'est évidemment pas neutre et lorsque le programme devient la notation de l'objet lui-même, la question de la qualité de ce s*ystème de notation opératoire* d'un point de vue cognitif devient primordiale : en quoi la description informatique, le programme, est une bonne notation du l'objet qu'il décrit.

1.1 Structuration des données et des programmes

Les langages de description musicale peuvent être grossièrement classés en deux catégories : les représentations déclaratives ou procédurales [Balaban 94]. Les langages déclaratifs décrivent la musique à l'aide de structures de données statiques, qui combinent généralement des éléments hiérarchiques avec un aspect temporel explicite. Dans l'approche procédurale, la dimension temporelle du résultat est un effet de bord du calcul. La représentation déclarative a beaucoup d'avantages, elle permet une meilleure compréhension de l'objet, est souvent plus simple

à manipuler. Elle correspond souvent à un mode de structuration usuel du compositeur, habitué à la dimension temporelle de la partition.

Pouvoir représenter l'aspect opératoire, c'est à dire habituellement le programme, de manière aussi simple est un problème difficile. Habituellement, le comportement du programme sur une portion temporelle particulière de l'objet musical ou sur une de ses dimensions (par exemple la hauteur) n'est pas forcément explicite dans le programme lui même, c'est seulement une conséquence de son évaluation.

Une des principale raison est que les outils conceptuels dont on dispose pour structurer les programmes ne sont pas aussi puissants que ceux que l'on utilise pour les données. La manière de décomposer un programme en parties plus simples dépend directement de la façon dont on pourra combiner les solutions [HUG 89].

1.2 Structuration de données agissantes

Beaucoup de langages de programmation, en particulier les langages fonctionnels, considèrent les fonctions comme des concepts de première classe. Les fonctions peuvent être passées en argument et retournées comme résultat, donc d'une certaine manière manipulées comme des données. Ceci permet de construire des fonctions d'ordre supérieur, un concept très puissant qui est un des points essentiel qui rend la programmation fonctionnelle si efficace : c'est la nouvelle *colle* qui donne des possibilités de modularisation beaucoup plus grandes, l'autre étant l'évaluation paresseuse [HUG 89]. Ce concept de modularisation par *agrégation de comportement séparés* (représentés sous formes d'objets) est d'ailleurs aussi largement répandu dans les techniques les plus récentes de développement objets qui font une utilisation très large de *pattern de classes réutilisables*, qui ressemble en de nombreux points à la notion de fonction d'ordre supérieur des langages fonctionnels.

La plupart des langages musicaux contiennent des opérations de structurations temporelles comme la *mise* en *séquence* ou la superposition *temporelle*. Mais habituellement ces opérations n'ont pas d'équivalent dans le domaine des programmes qui n'ont pas dimension de temporelle explicite. Ainsi une structure de données combinant des fonctions ne peut pas être utilisée comme une fonction ordinaire, c'est à dire appliquée à une argument. Autrement dit, les opérations de structuration qui marchent pour les données ne fonctionnent pas pour les fonctions.

D'où l'idée de donner un *caractère opératoire à des structures hiérarchiques de processus* en particulier aux structures construites avec les opérations habituelles de structuration temporelles comme la mise en séquence ou le mixage. C'est une des particularité d'Elody où les processus peuvent être composés comme les objets musicaux classiques et se comporter, vu comme un tout, comme des nouveaux processus qui combinent les comportements de leur éléments.

2. Le modèle opératoire d'Elody

Dans Elody, on évite la séparation habituelle entre le langage de description du domaine musical et le langage de programmation. En effet, les capacités de programmation sont directement inscrites dans le langage musical. Elody comporte les éléments suivants :

• des notes et des silences avec leur attributs habituels: hauteur, vélocité, timbre et durée.

• des opérations de structuration temporelle: SEQ : mise en séquence dans le temps, MIX : superposition dans le temps, EXPAND : compression/dilatation, BEGIN , END : découpage d'un objet.

• des opérations de transformations des paramètres de hauteurs (par addition de demi-tons), de vélocité (par addition d'un offset de vélocité), de durée (par multiplication) et de changement de timbre (transposition de canal).

• les opérations d'abstraction et d'application du lambda-calcul.

Une structure Elody est représentée par un arbre, dont les nœuds sont des constructeurs du langage et les feuilles des éléments primitifs. Les opérations représentées par les constructeurs ne sont effectuées que lors de la phase de rendu (lorsque l'objet est joué ou affiché graphiquement) et l'arbre n'est jamais modifié. Ainsi chaque objet contient toujours l'histoire de sa construction. C'est cette caractéristique qui devient si importante lors du processus d'abstraction. Ainsi si on sélectionne par exemple la deuxième note d'une séquence, cette note conserve son statut de deuxième note d'une séquence, et si maintenant la séquence originale est rendue variable

dans cette note, on définit ainsi le concept *prendre la deuxième note d'une séquence* qui représente d'une certaine manière l'idée de degré.

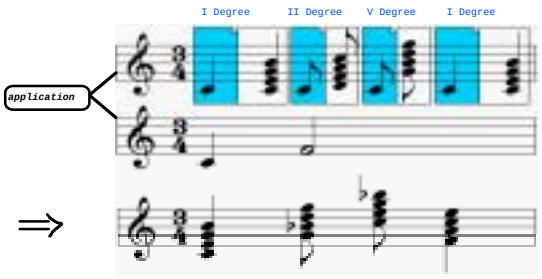
Les abstractions dans Elody sont des *structures musicales généralisées avec des parties variables*. Elles sont définies à partir de structures concrètes et peuvent être vues comme des classes d'objets, pourront être visualisées et écoutées comme des prototypes de la classe d'objet qu'elle représentent et se comporter comme des fonctions lorsqu'elles seront appliquées sur des arguments pour produire de nouveaux résultats. Comme elles sont définies à partir d'objets concrets, les abstractions restent des objets musicaux, elles ont une durée, peuvent être manipulées comme les objets habituels c'est à dire avec les opérations temporelles : SEQ et MIX et les transformations : transposition, dilatation et compression ... etc.

2.1 La sémantique opérationnelle des partitions de processus

Le caractère opératoire du langage est défini par un ensemble de règles et une stratégie de réduction qui permettent de passer d'un terme en intention à un terme en extension. En plus de la notion de bêta réduction du lambda-calcul, des règles de réduction supplémentaires donnent un *caractère opératoire* aux constructions temporelles d'abstractions. Par exemple lorsqu'une séquence d'abstractions est appliquée à un argument, chaque abstraction s'applique à la portion temporelle correspondante de l'argument.

Ajouter cette sémantique au langage donne une plus grande homogénéité aux opérations de structuration puisque les fonctions deviennent composables comme les données, ainsi il devient possible d'utiliser les mêmes outils de manipulation temporelle sur les données et sur les processus.

L'autre conséquence est qu'un certain nombre de constructions sont plus faciles à exprimer de cette manière. Ainsi, le concept de partition, naturel pour exprimer la composition dans le temps de structure musicales, peut être aussi utilisé pour composer des abstractions. Dans une structure musicale, il est souvent naturel d'exprimer un comportement ou un processus de transformation qui opère sur une tranche temporelle de l'objet Disposer d'un langage avec une sémantique applicative pour les partitions de processus permet d'exprimer ce concept comme l'application d'une *partition de processus* où la transformation désirée est placée dans le temps à l'endroit où elle devra s'appliquer sur la *partition argument.* De même, il sera possible d'exprimer l'enchaînement temporel de deux transformations comme la mise en séquence de ces deux transformations. Le résultat sera une nouvelle fonction dont l'application sur un argument enchaîne la séquence des résultats des applications de chaque transformation sur des portions successives de l'argument. Voici un exemple d'application temporelle d'une séquence d'abstractions, ici des accords abstraits (ou intentionnels) où la tonique a été rendue variable, sur une séquence de notes, chaque accord s'appliquant sur la portion temporelle de l'argument qui correspond à sa durée.



Dans cet exemple apparaît clairement la vision multiple que l'on a d'un même objet : les accords du I, II, V degrés vus comme des accords intentionnels sont manipulés comme des données lorsqu'on les compose en séquence par exemple. Si on écoute cette séquence, on entend la suite des accords prototypes qui ont servi a construire chaque abstraction, l'accord abstrait se comporte lors du rendu, comme un *prototype de la classe* des objets qu'il représente. Ici on entend les accords réalisés Do Maj7 (prototype de la classe des accords Maj7), Ré min7 et G7. Enfin, chaque accord est aussi une fonction capable de produire un accord réalisé à partir d'une tonique et la séquence des trois abstractions se comporte comme un nouveau processus capable de produire une séquence d'accords réalisés lorsqu'il est appliqué à une séquence de toniques.

2.2 La construction par "couches"

La sémantique applicative permet aussi de construire des objets musicaux dans lesquels des éléments différents sont exprimés par des couches d'abstractions montées en séquence qui seront appliquées les unes sur les autres pour donner le résultat final. Chaque application partielle donne un résultat intermédiaire qui peut être écouté et qui est explicatif d'un aspect de la structure finale.

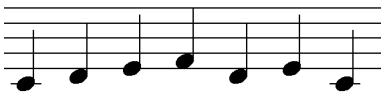
3. Modélisation de notions de la musique tonale

La construction de représentations de la connaissance musicale est souvent une étape première dans l'élaboration d'environnements de composition ou d'analyse. Plusieurs systèmes ont été développés comme MusES, un système objet de représentation opérationnelle des notions de notes, intervalles, gammes et accords, utilisé dans différentes applications d'analyse harmonique ou d'harmonisation automatique [PRC 1996]. D'autres systèmes sont basés sur le paradigme de satisfaction de contraintes, comme Situation, un langage visuel basé sur les contraintes pour la composition musicale [BR 98], ou encore sur la programmation fonctionnelle comme Haskore basé sur le langage Haskell, où des propriétés algébriques peuvent être vérifiées sur les structures musicales [HMGW 96]. Chaque système est alors souvent dépendant de l'objectif ou de la classe d'application visée.
Dans Elody, c'est en partant d'objets musicaux simples, et avec les opérations d'abstraction et d'application que des notions musicales plus élaborées peuvent être décrites. Une abstraction va être considérée suivant le point de vue, soit comme représentant une classe d'objets, soit comme une fonction capable de produire les différentes instances de cette classe. Le but est de représenter les connaissances musicales nécessaires à la construction de structures tonales sous la forme de représentations manipulables, composables et écoutables. On va construire des concepts qui *contiennent dans leur structure même les éléments de leur manipulation ultérieure.*

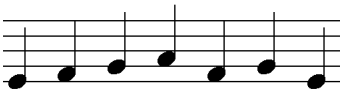
3.1 Un exemple de transformation tonale

Dans cet exemple, nous introduisons une manière d'exprimer quelques transformations tonales. Tout d'abord nous construisons une gamme de Do majeur vue comme une séquence de notes simples. Ensuite, nous découpons les différents degrés de cette gamme en utilisant les constructeurs Elody adéquats : opérateurs BEGIN et END. Les degrés ainsi définis contiennent donc la gamme de base qui peut être rendue variable. Abstraire la gamme d'un degré conserve alors le processus de couper une note dans la gamme à la position correspondante. Appliquer cette abstraction à une gamme différente revient à appliquer le processus de couper à cette nouvelle gamme et produira une note *ayant le même statut tonal dans cette nouvelle gamme*.

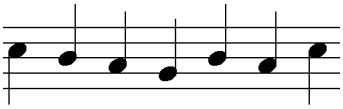
En utilisant les degrés précédemment définis, on peut construire la mélodie suivante :



On rend maintenant variable la gamme majeure utilisée pour la construction des degrés dans la mélodie. Appliquée à des gammes différentes, la fonction obtenue produira différentes transformations tonales. Par exemple en appliquant la fonction sur le mode III de Do majeur c'est à dire E, F, G, A ,B, C, D on obtient la transposition tonale d'une tierce :



Appliquée à une gamme de Do majeur descendante, la fonction produira le miroir du motif mélodique :



3.2 Construction d'une librairie tonale

En partant des idées exposées précédemment nous allons élaborer une représentation des concepts de la musiques tonales plus complexes. Le principe fondamental est de construire les connaissances musicales sous forme hiérarchique et par couches, en partant d'un nombre d'éléments réduit et par des actions de généralisation/spécialisation de bâtir les concepts de plus haut niveau. A partir d'un objet concret, on définit une classe d'objet par abstraction, et la fonction obtenue pourra être utilisée pour engendrer tout un ensemble de nouveaux objets concrets. On établit ainsi des relations entre des objets par le fait qu'ils sont construits avec des éléments communs. Toute forme construite dans Elody est représentée par un arbre qui *conserve l'histoire de sa*

construction, qui pourra être modifiée par la suite. C'est de cette façon, en *reconstruisant l'histoire de l'objet avec des ingrédients différents* que les transformations seront exprimées.

C'est un modèle sous la forme d'un arbre avec une racine commune à tous les objets. A chaque niveau de l'arbre, la variation d'un des éléments se *transmet* à tous les concepts qui en dépendent. Un objet feuille comporte autant de facettes que d'étapes qui ont permis de le construire. Toutes ces étapes restent accessibles. Chaque abstraction intermédiaire est un objet musical écoutable qui se comporte comme un prototype du concept qu'il représente, on peut donc *vérifier que les concepts fonctionnent bien*.

Cette hiérarchie de concepts n'a pas la vocation d'être complète, elle modélise une partie des notions de la musique tonale à partir des étapes suivantes :

1°) Construction de la gamme chromatique de 12 notes commençant sur un Do sous la forme d'une séquence construite avec l'opérateur SEQ du langage.

2°) Définition des *degrés chromatiques*, vus comme des index sur la gamme. Cette notion est modélisée en utilisant les opérateurs BEGIN et END du langage qui permettent de découper à l'intérieur d'un objet musical en établissant une relation entre les notes individuelles et la gamme chromatique dont elles sont issues.

En rendant variable la gamme chromatique dans un degré, on obtient l'opération d'indexation capable de sectionner le degré correspondant dans la gamme chromatique.

3°) Construction de la gamme chromatique rétrograde.

4°) A partir des degrés chromatiques, construction de la gamme majeure de 7 tons et de différentes gammes mineures : mineure harmonique et mineure mélodique ascendante.

En rendant variable la gamme chromatique dans une des gammes de 7 degrés, on obtient l'opération qui construit la gamme de 7 degrés à partir d'une gamme de 12 degrés.

5°) Construction de la gamme relative mineure comme une opération effectuée sur la gamme majeure.

En rendant variable la gamme majeure dans sa relative mineure, on obtient l'opération qui construit la relative mineure à partir d'une gamme majeure donnée.

6°) Construction des modes, vus comme des permutations circulaires de la gamme majeure, ainsi que du rétrograde de la gamme majeure.

En rendant variable la gamme majeure dans chaque mode, on obtient la fonction qui calcule les différents modes pour une gamme donnée. Ces fonctions peuvent être utilisées par exemple pour engendrer les 7 modes des gammes mineures.

7°) Construction des *degrés diatoniques* sur la gamme majeure par découpage à l'aide des opérateurs BEGIN et END des 7 notes de la gamme majeure.

En rendant variable la gamme majeure dans chaque degré diatonique, on obtient une opération d'indexation capable de sélectionner le degré correspondant dans la gamme diatonique.

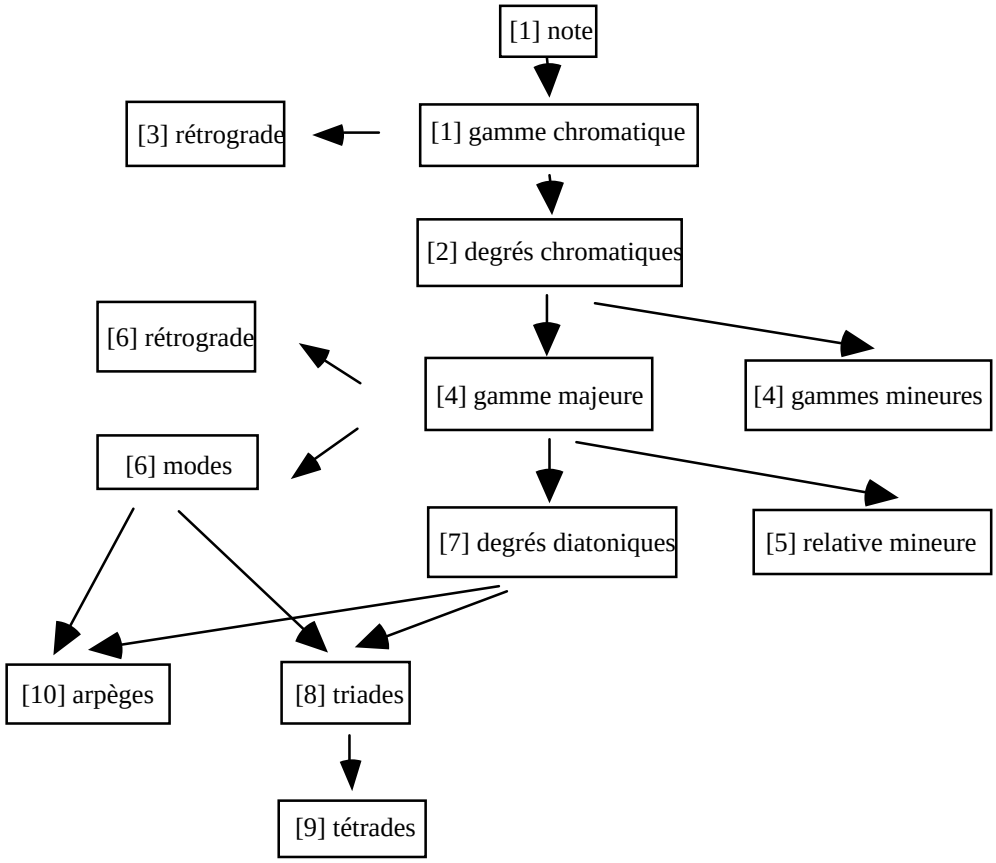
8°) Définition des triades par construction de la triade des degrés diatoniques I,III,V. C'est un accord majeur concret Do, Mi, Sol. En rendant variable la gamme majeure, on obtient la fonction qui engendre toutes les triades. En appliquant cette fonction sur les 7 modes de la gamme majeure, on obtient tous les accords diatoniques en position fondamentale. On modélise ici la succession des accords diatoniques comme un accord du I degré sur les différents modes de la gamme.

En rendant variable la gamme majeure dans chaque triade, on obtient des fonctions qui construisent les triades diatoniques à partir d'une gamme diatonique. Ces fonctions peuvent être utilisées par exemple pour engendrer les 7 triades des gammes mineures.

9°) Construction des tétrades en ajoutant le degré VII à la triade I,III,V. En rendant variable la gamme majeure, on obtient la fonction qui engendre toutes les tétrades. En appliquant cette fonction sur les 7 modes de la gamme majeure, on obtient tous les accords diatoniques de 4 sons en position fondamentale.

En rendant variable la gamme majeure dans chaque tétrade, on obtient des fonctions qui construisent les tétrades diatoniques à partir d'une gamme diatonique. Ces fonctions peuvent être utilisées par exemple pour engendrer les 7 tétrades des gammes mineures.

10°) Construction des arpèges de degrés sur le même principe par construction explicite de l'arpège du premier degré et engendrement des autres arpèges en utilisant les modes.



3.3 Quelques opérations tonales

Garder l'histoire d'une objet construit à partir des éléments définis précédemment permet de manipuler les différentes étapes de sa construction, d'accéder indépendamment à ses multiples facettes. Un certain nombre d'opérations classiques de la musique tonale s'expriment alors comme le remplacement, par abstraction/application, d'un ingrédient par un autre et *reconstruction de l'histoire avec le nouvel ingrédient*.

Un objet musical construit en utilisant comme matériaux des degrés et accords diatoniques (au sens défini précédemment, c'est à dire construit avec des degrés diatoniques) contient dans sa structure la ou les gammes utilisées pour construire les degrés et les accords. Il est alors possible de réaliser les opérations suivantes :

- transposition tonale : remplacement de la gamme par un mode de la gamme. Par exemple transposer tonalement d'une quinte revient à remplacer dans l'objet la gamme utilisée par le mode du V degré de la gamme.

- passage en mineur : remplacement de la gamme majeure par une gamme mineure.

- miroir diatonique : remplacement de la gamme majeure par le rétrograde de la gamme majeure.

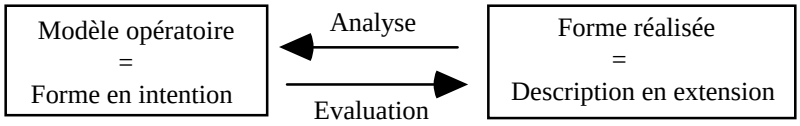
- miroir chromatique : remplacement de la gamme chromatique par le rétrograde de la gamme chromatique.

- combinaison de ces opérations : par exemple en remplaçant dans l'objet la gamme par le mode III d'une gamme mineure, on réalise un passage en mineur avec une transposition tonale d'une tierce.

Il est possible d'agir de manière globale sur des objets ayant des caractéristiques en commun mais dont la forme réalisée est différente. Sur une séquence d'accords de degrés réalisés sur des gammes différentes, donc qui *sonnent différemment*, on peut agir sur les accords d'un degré particulier et réaliser par exemple une opération globale de substitution d'accords. A contrario, des objets réalisés identiques mais dont la forme intentionnelle est différente restent manipulables de manière indépendante. Par exemple l'accord réalisé Fa, La, Do à (au moins) deux formes intentionnelles différentes : accord du I degré en Fa ou accord du IV degré en Do.

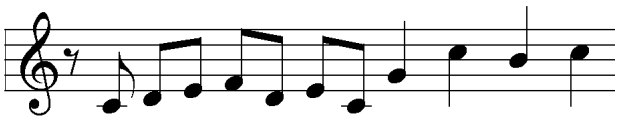
4. Utilisation pour l'analyse et la modélisation : un modèle opératoire d'une pièce tonale.

On entend ici par *modèle opératoire* une *description structurelle* d'une pièce qui fait apparaître explicitement les *processus de composition* utilisés avec leur évolution temporelle et dont la *forme réalisée* c'est à dire le résultat après évaluation est semblable à la partition originale. L'intérêt pour l'analyste ou le compositeur est de comprendre et manipuler l'objet musical en tant que combinaison de processus de composition, la forme en intention, en laissant au système le soin de calculer la forme réalisée (en extension). Le modèle, parce qu'il rend compte d'un certain nombre d'aspects, en laissant de coté d'autres, se prête ainsi à la réalisation de variantes.



Dans un cadre tonal, on va définir une structure musicale à partir des éléments de modélisation du système tonal définis précédemment, sous la forme de plusieurs concepts séparés : tonalité, structure harmonique, structure des hauteurs, structure rythmique, etc. La finesse de la modélisation conditionnera ensuite le type de manipulation que l'on pourra faire sur l'objet construit rythmique.

La pièce étudiée ici est l'invention n°1 de Bach en Do majeur et seulement les 4 premières mesures ont été modélisées. L'invention, une forme classique de la musique tonale, est construite à partir d'un motif qui va être exposé et développé au cours de la pièce en utilisant différentes techniques d'écriture dans un contexte tonal : réécriture du motif sur différents degrés de la gamme, dans d'autres tonalités ou modes, augmentation, miroir..., et combinaison de ces différentes techniques. Les motifs utilisés sont les suivants :



Motif A Motif B

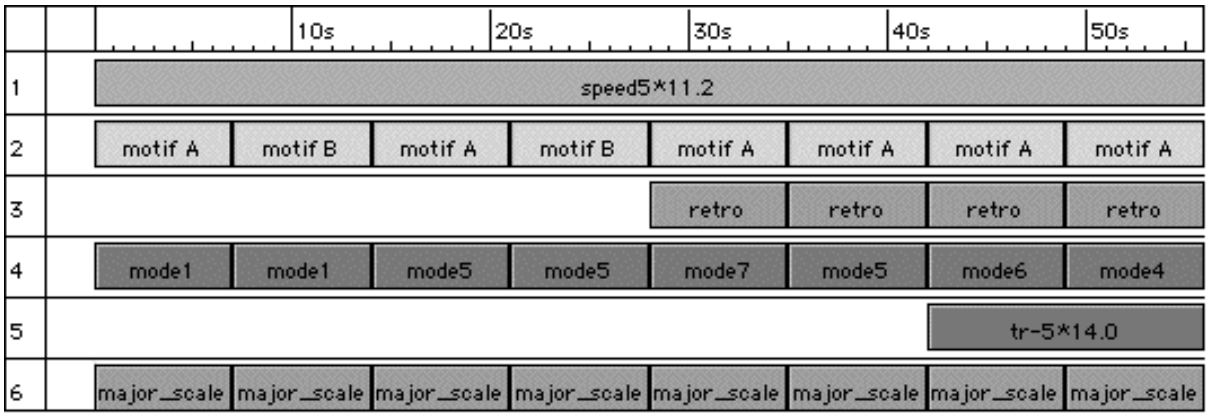
4.1 Les niveaux d'analyse

La pièce a été analysée à plusieurs niveaux :
- l'évolution des tonalités
- l'évolution harmonique sous la forme de la succession des degrés à l'intérieur d'une gamme
- les transformations particulières, ici miroir et augmentation
- la structure des hauteurs, c'est à dire l'évolution des motifs
- la structuration rythmique n'est ici pas prise en compte, on considère en effet que les motifs de base contiennent déjà leur structure rythmique.

4.2 Le modèle

A partir des notions tonales construites précédemment, l'invention en Do majeur est réalisée dans l'éditeur TimeLine d'Elody où les parties de main gauche et main droite sont décrites par des ensembles des pistes différentes. C'est une partition qui reprend les éléments principaux de l'analyse et qui organise les différents concepts, évolution des hauteurs, tonalité, progression harmonique, transformations diverses... sous la forme de processus organisés dans le temps en plusieurs couches qui s'appliquent les unes sur les autres. Comme dans tout

modèle, des choix ont été fait pour décider les concepts pertinents à exprimer. La partie de la main droite a été définie de la façon suivante :



- la piste 2 pour l'évolution des hauteurs à partir de deux motifs primitifs construits explicitement comme des *mélodies de degrés* : motif A et motif B. Cette piste sera appliquée sur une structure harmonique elle même décomposée en 4 pistes.

- les pistes 5 et 6 représentent l'évolution temporelle des tonalités : séquences des gammes, et construction d'autres gammes par transposition: les 3 premières mesures sont en Do majeur, la dernière en Sol majeur, construite comme une transposition chromatique d'une quinte de la gamme de Do majeur.

- la piste 4 décrit l'évolution harmonique. Elle est construite par l'application de fonctions *mode* sur la séquence des gammes majeures définie précédemment. La fonction *mode n* calcule le n^{ième} mode d'une gamme donnée avec pour la 1° mesure : mode I , pour la 2° mesure : mode V, qui décrit une *transposition tonale* d'une quinte, pour les 3° et 4° mesures : marche harmonique descendante par tierces, décrites par la succession des mode 7 et mode 5 sur la gamme de Do majeur, mode 6 et mode 4 sur la gamme de Sol majeur.

- la piste 3 décrit un calcul de miroir sur les mesures 3 et 4. Dans ce modèle on construit le miroir du motif comme l'application du motif au *rétrograde du mode*, construit par l'application d'une *fonction rétrograde* sur la séquence des modes définis précédemment.

- une piste 1 pour le tempo global, ici une contraction temporelle d'un facteur 2.

Le modèle de la main gauche peut être construit de la même manière, toute la structure harmonique est identique, à une transposition d'une octave près. La piste 3 est remplacée par des fonctions d'augmentation et la piste 2 par une autre séquence des motifs.

4.3 Écoute incrémentale

Les différentes pistes s'appliquent les unes sur les autres. La piste 5 s'applique sur la piste 6, la piste 4 sur le résultat précédent et ainsi de suite. La compréhension du modèle de la pièce est facilitée par la possibilité d'une *écoute incrémentale* où les différents éléments de la structure apparaissent progressivement lorsque les pistes sont activées. Plusieurs constructions progressives sont possibles suivant le choix que l'on fait dans l'ordre d'activation des pistes, par exemple :

- couche 1 : évolution des tonalités, succession de gammes de Do majeur sur 3 mesures puis Sol majeur sur 1 mesure.

- couche 2 : évolution harmonique, les fonctions modes appliquées à une gamme rendent en résultat le mode correspondant de la gamme. On entendra donc la séquence :

M1 de Do	M1 de Do	M5 de Do	M5 de Do	M7 de Do	M5 de Do	M6 de Sol	M4 de Sol

- couche 3 : la fonction miroir rend en résultat le rétrograde de la gamme ou mode sur laquelle elle est appliquée. L'application de la couche 3 sur le résultat précédent produit l'évolution des modes avec leur forme miroir.

M1 de Do	M1 de Do	M5 de Do	M5 de Do	R [M7 de Do]	R [M5 de Do]	R [M6 de Sol]	R [M4 de Sol]

- couche 4 : applications du motif sous la forme d'une *mélodie de degrés diatoniques* sur la structure précédente.

- couche 5 : enfin la structure temporelle globale avec une accélération d'un facteur 2.

Ce modèle se prête à l'étude de toute une série de variations. Comme tous les ingrédients utilisés : motifs, gammes, fonctions mode, miroir, peuvent être rendus variable, il est possible d'écouter tout un ensemble de variantes : passages sur des tonalités mineures, changement du motif, etc.

5. Conclusion

Elody est un environnement de composition proposant un modèle de programmation original basé sur l'extension d'un langage musical avec les concepts du lambda-calcul, ou les programmes sont définis à partir de structures concrètes. Nous avons vu comment il est possible de donner aux abstractions un statut d'objet musical "normal", puisqu'on peut les écouter, les manipuler avec les opérations utilisées sur les objets musicaux habituels, et ainsi les percevoir de façon multiple, comme données ou comme processus.
Nous avons montré comment des notions opératoires de plus haut niveau peuvent être décrites, et de quelle manière, après avoir défini une description hiérarchique de concepts, il est possible de modéliser un certain nombre de transformations tonales classiques.
Nous avons vu enfin comment la sémantique opérationnelle donnée aux partitions de processus organisés dans le temps, permet de modéliser de manière élégante des pièces tonales. La structuration temporelle des opérations de composition est rendue explicite. La structure globale peut être instanciée et écoutée de manière progressive.
Ce travail a été réalisé dans une étape de validation de l'environnement Elody, et bien que les exemples soient issus plutôt d'une démarche d'analyse et de modélisation, habituellement inverse de la composition, nous espérons qu'ils montrent clairement les conséquences intéressantes de l'approche globale.

Références

[Agon 98] Agon .C, OpenMusic: un langage visuel pour la composition musicale assistée par ordinateur. Thèse de doctorat de l'Université de Paris 6. IRCAM, 1998

[Balaban 94], Balaban M., “Introducing Formal Processing into Music - The Music Structure Approach”, *Technical Report FC-94-07*, Dept. of Math. and Comp. Science, Ben-Gurion University of the Negev, 1994.

[BR 98] Bonnet A., Rueda C., "Situation : un langage visuel basé sur les contraintes pour la composition musicale" Recherches et Applications en Informatique Musicale, p 65-74, Edition Hermes, 1998

[Dannenberg 89], Dannenberg R. B., “The Canon Score Language”, *Computer Music Journal*, 13 (1), pp. 47-56, 1989.

[HMGW 96] Hudak P., T. Makucevich, S. Gadde, B. Whong, “Haskore Music Notation - An Algebra of Music”, Journal of Functional Programming, Cambridge University Press, 6(3), June 1996.

[HUG 89] Hughes J., "Why Functional Programming Matters" Computer Journal 32!2) 1989

[Oppenheim 1989] Oppenheim .D, "DMIX: An Environment for Composition " Proceedings of the ICMC 1989, Computer Music Association, San Francisco, pp.226-233

[OFL 97] Orlarey.Y, Fober.D, Letz S, "L'environnement de composition musicale Elody", Actes des JIM97, Lyon 1997

[PRC 96] Pachet F., Ramalho G., Carrive J. ,"Repensenting temporal musical objetcs and reasoning in the MusES system". Journal of New Music Research ,vol 25 n°3 , pp 252-275, 1996