



HAL
open science

De Boxes à Iscore : vers une écriture de l'interaction

Antoine Allombert, Myriam Desainte-Catherine, Gérard Assayag

► **To cite this version:**

Antoine Allombert, Myriam Desainte-Catherine, Gérard Assayag. De Boxes à Iscore : vers une écriture de l'interaction. Journées d'Informatique Musicale, Mar 2008, Albi, France. pp.79-83. hal-00353628

HAL Id: hal-00353628

<https://hal.science/hal-00353628>

Submitted on 15 Jan 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

DE BOXES À ISCORE : VERS UNE ECRITURE DE L'INTERACTION

Antoine Allombert
LaBRI/IRCAM
allomber@labri.fr

Myriam Desainte-Catherine
LaBRI
myriam@labri.fr

G rard Assayag
IRCAM
assayag@ircam.fr

ABSTRACT

Nous pr sentons dans cet article les d veloppements actuels des travaux de recherche men s au SCRIME (Studio de Cr ation et de Recherche en Informatique et Musique  lectro-acoustique) visant   aboutir   un syst me de partitions interactives pour la composition et l'interpr tation bas  sur des contraintes temporelles. Ces travaux effectu s en collaboration avec l'Ircam font suite   ceux d'Anthony Beuriv  et visent   cr er une descendance au logiciel *Boxes*. L'impl mentation actuelle de cet outil est double, d'une part elle prend la forme d'une extension des *Maquettes* d'*OpenMusic*, mais  galement d'une version ind pendante qui reste cependant incompl te. La poursuite de ces travaux est associ e au projet Virage dont l'objectif est la cr ation d'un outil interactif pour la r gie num rique de spectacles vivants.

1. INTRODUCTION

Le logiciel *Boxes* initialement d velopp  au SCRIME par Anthony Beuriv  et dont on peut trouver une description dans [3] est une tentative d'approche originale d'un environnement de composition assist e par ordinateur en m lant mod le spectral, structures hi rarchiques et contraintes. Dans sa forme, il est assez proche du syst me des *Maquettes* d'*OpenMusic* [2] en ce sens qu'il repr sente les sons composant la partition sous forme de "boites" (d'o  son nom) positionn es sur une "feuille blanche" repr sent e par la fen tre d' dition. La figure 1 pr sente un exemple de partition *Boxes*. Dans cet environnement, le temps  volue en abscisse, de gauche   droite ; l'axe des ordonn es n'a pas de signification particuli re, il n'est pas organis  pas pistes.

Comme on peut l'observer sur la figure 1, un des fondements de *Boxes* est de s'appuyer sur un mod le hi rarchique. Le compositeur a ainsi la possibilit  de d finir des structures de niveaux diff rents et de les inclure les uns dans les autres. Il peut ainsi regrouper des sons dans des structures telles que des accords ou des lignes m lodiques par exemple, une boite contenant d'autres boites pouvant  tre elle-m me incluse dans une boite de plus haut niveau. Les seules restrictions de cette repr sentation hi rarchique  tant qu'une boite ne peut  tre incluse dans elle-m me ni dans un de ses composants. En outre, une boite pouvant  tre incluse dans plusieurs boites de plus haut niveau, on obtient au final une structure de graphe orient  acyclique.

Boxes s'appuie  galement sur un syst me de contraintes

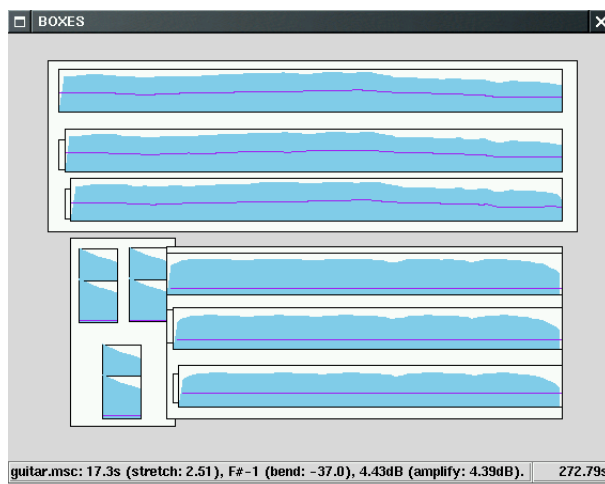


Figure 1. Un exemple de partition *Boxes*

en s'inspirant de l'utilisation qu'en fait le logiciel *Music-Space* [7] pour la spatialisation de sources sonores. Ainsi, le compositeur peut contraindre l'agencement des boites au moyen de relations binaires dites de "Allen" pr sent es sur la figure 2. Celles-ci permettent donc au compositeur de d finir une coh rence temporelle pour sa pi ce qui sera maintenue tout au long du processus de composition. Ainsi apr s introduction de contraintes dans la partition, lors du d placement ou du changement de la taille d'une boite par l'utilisateur, le syst me se charge de r ajuster la position ou la taille des autres boites si besoin est pour maintenir la validit  des contraintes pos es par le compositeur. Il est important de noter que le syst me inf re  galement des contraintes implicites qui sont :

- la fin d'une boite doit toujours se produire apr s son d but
- une relation *during* entre une boite hi rarchique et chacune des boites qui la composent

Pour calculer les nouvelles valeurs des positions et dur es des boites apr s une modification de l'utilisateur, *Boxes* utilise la biblioth que Cassowary impl mentant l'algorithme du simplexe pour minimiser une fonction objective prenant en compte d'une part les contraintes de Allen, les contraintes implicites et des contraintes de comportement pour orienter le calcul et  viter d'obtenir des nouvelles valeurs trop  loign es des valeurs d'origine. Par ce m canisme, le compositeur peut d finir dans un premier temps la coh rence temporelle de sa pi ce avant d'affiner la position

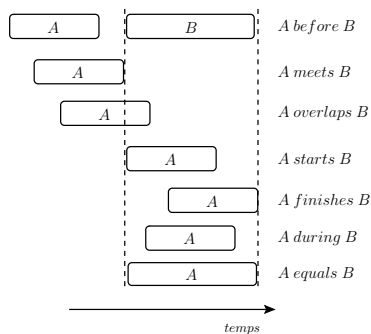


Figure 2. Les relations de Allen

de chacune des boîtes tout en étant assuré que ses modifications ne remettons pas en cause l'organisation globale qu'il a définie.

Enfin, pour synthétiser les sons, *Boxes* utilise un modèle spectral SAS [5]. Ce type de modèle permet certaines transformations des sons, notamment l'étirement sans artefact fort utile en cas de modification de la taille d'une boîte ; transformations aux quelles le modèle temporel ne donne pas accès.

Le développement de *BOXES* a été momentanément interrompu. En outre ce logiciel a pour but d'être et de rester un logiciel libre, la bibliothèque Cassowary devenant propriétaire une difficulté supplémentaire s'est greffée sur la poursuite du projet.

Cependant, la reprise des travaux théoriques autour du modèle et la volonté de l'étendre à la création de partitions interactives a été l'occasion de reprendre le développement de cet outil.

2. MODÈLE THÉORIQUE

Comme annoncé précédemment, les travaux que nous présentons ici sont le prolongement de ceux qui ont mené à la première mouture de *Boxes*. Ainsi, nous reprenons deux de ses aspects fondamentaux : le modèle hiérarchique et l'utilisation des relations de Allen.

Cependant nous avons cherché à l'enrichir pour permettre la création de partitions "interprétables". En effet, l'évolution des techniques et de la musique au $XX^{ième}$ siècle et notamment l'avènement de la musique électroacoustique ont conduit les compositeurs à créer des pièces sur support, non jouables autrement que par leur diffusion sur des systèmes d'écoute, la part d'interprétation lors des représentations étant dévolue à la spatialisation. Cette situation a privé une partie des œuvres modernes de l'apport de l'interprétation dont a toujours profité la musique instrumentale. Notre objectif est donc de développer un système permettant l'interprétation de pièces électroacoustiques au travers des mêmes vecteurs que celle des pièces instrumentales.

Nous nous appuyons sur les travaux de Jean Haury [6] qui identifie 4 possibilités pour l'interprétation :

- les variations dynamiques

- l'accentuation
- l'articulation
- les modifications agogiques (changement des dates de début et de fin des notes)

Dans le cadre de notre étude, nous ne nous intéressons qu'aux modifications agogiques. Par conséquent, nous délaissions le contenu des boîtes et nous considérons celles-ci d'un point de vue uniquement symbolique.

Jean Haury précise également que le musicien a accès aux possibilités de l'interprétation à travers des points de contrôle placés dans la pièce qu'il appelle "points d'interaction". Un point d'orgue est un bon exemple de point d'interaction pour la musique instrumentale dans la mesure où le musicien ou chef d'orchestre peut choisir la durée du point d'orgue.

En outre, ces possibilités d'interaction et donc de modification de la pièce s'accompagnent de la définition par le compositeur d'un cadre dans lequel celles-ci vont pouvoir s'exprimer. Dans la musique instrumentale, ce cadre est fixé au travers d'indications du compositeur par exemple de volume (*p*, *ff*...) ou de tempo (*accelerando*...). Ainsi l'interprète est amené à jouir d'un certain nombre de libertés laissées par le compositeur tout en restant dans un cadre fixé par ce même compositeur. Comme nous nous restreignons aux modifications agogiques et donc à la possibilité de décaler des notes ou d'en modifier la durée par rapport à ce qui est écrit par le compositeur, celui-ci doit être en mesure de définir une organisation temporelle de sa pièce qu'il souhaite voir respectée quels que soient les choix de l'interprète. On voit ici se dessiner la cohérence temporelle évoquée dans l'introduction et que *Boxes* permet de définir grâce aux relations de Allen. Nous étendons donc l'utilisation de ces relations depuis l'aide à la composition statique vers la définition du cadre temporelle dans lequel évoluera l'interprète lors de l'exécution.

Notre modèle présente les caractéristiques suivantes :

- la représentation par boîtes hiérarchiques
- l'utilisation des relations de Allen
- l'introduction de points d'interaction sur les débuts et fins de boîtes.
- une représentation générique des partitions permettant l'utilisation d'une même machine d'exécution pour toutes les pièces et non pas de systèmes "ad hoc" pour chaque pièce

La figure 3 présente une partition interactive. Dans cet exemple assez simple, une boîte hiérarchique T_1 contient 4 boîtes simples T_2 , T_3 , T_4 et T_5 dont certaines sont contraintes entre elles par des relations de Allen, le début de T_4 étant rendu interactif par l'ajout d'un point d'interaction T .

Il est important de noter que l'outil que nous cherchons à développer utilisant ce modèle devra à la fois permettre l'édition de partitions interactives par le compositeur en lui donnant accès aux différents éléments exposés précédemment, mais devra également être capable d'exécuter ces partitions.

Pour ce qui concerne la partie "édition", nous utilisons les mêmes mécanismes que *Boxes* en nous appuyant sur

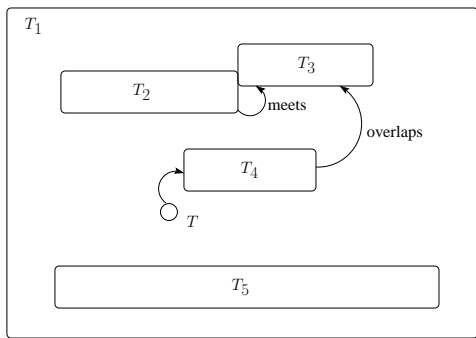


Figure 3. Un exemple de partition

un système de contraintes à résoudre chaque fois qu’une modification intervient. Pour la partie “exécution” il nous est apparu qu’une solution basée sur une résolution du système de contraintes dès que l’interprète modifie la valeur d’une date de début ou de fin de note risquait de conduire à des temps de calcul trop longs pour le temps réel. Ainsi, nous avons cherché à obtenir une représentation des partitions exécutée par la machine générique qui code les relations de Allen de telle manière qu’aucun calcul ne soit nécessaire en temps réel pour maintenir ces relations. Nous avons choisi les réseaux de Pétri qui gèrent des processus concurrents tout en permettant leur synchronisation à des moments précis, soit exactement ce que nous désirons. Pour permettre l’interprétation, nous transformons la partition en réseau de Pétri qui sera l’objet effectivement exécuté.

Pour des détails plus précis sur notre modèle théorique et notamment sur l’utilisation que nous faisons des réseaux de Pétri, le lecteur pourra se reporter à [1].

Afin de marquer l’évolution depuis *Boxes* et de mettre en lumière les possibilités d’interaction de ce système, nous avons choisi d’appeler celui-ci *Iscore* pour “interactive score”.

3. IMPLÉMENTATION

3.1. Partition statique

Actuellement, il existe deux implémentations de *Iscore*, l’une développée au LaBRI par Bruno Valèze, l’autre prenant place dans le système des *Maquettes* d’*OpenMusic*; cette dernière version étant la plus aboutie, nous présentons cette implémentation.

Dans *OpenMusic*, les *Maquettes* permettent d’organiser dans le temps les sons obtenus par évaluation des Patches. Il s’agit donc d’un véritable environnement dans lequel le compositeur va venir placer ses patches représentés sous formes de boîtes sur une “feuille blanche”, l’axe des abscisses représentant le temps de gauche à droite, l’axe des ordonnées n’ayant pas de sens a priori. L’évaluation de la maquette produira une pièce sonore dans laquelle les sons produits par chaque patch qui la compose interviendront aux dates représentées par les positions des

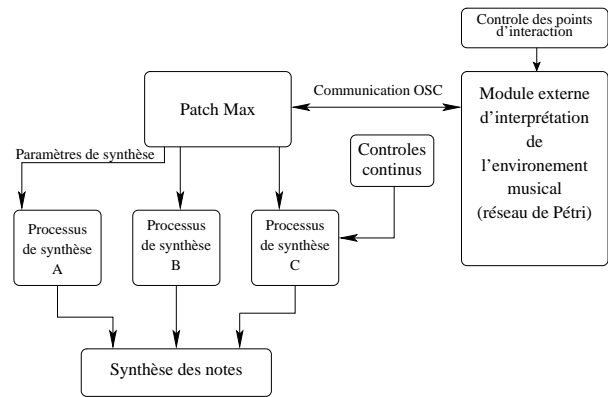


Figure 4. Architecture du système

patches dans la Maquette. On voit bien ici une certaine similitude avec *Boxes* évoquée en introduction.

Comme nous nous intéressons uniquement à l’aspect symbolique des boîtes, nous avons utilisé l’interface graphique des *Maquettes* pour créer avec l’aide de Jean Bresson de l’équipe Représentations Musicales de l’Ircam des objets boîtes entre lesquelles l’utilisateur va pouvoir poser des relations de Allen. La figure 5 montre un exemple de création de partition en présentant l’interface permettant créer les boîtes et de gérer les relations de Allen posée entre celles-ci.

Comme dans *Boxes*, pour la partie édition, nous devons résoudre un système de contraintes impliquant les dates de début et de fin des boîtes à chaque modification effectuée (ajout d’une relation, déplacement/étirement d’une boîte). Cette opération est déléguée à un solveur de contraintes appelé *Gecode* développé par Christian Schulte [8].

3.2. Interaction

Naturellement l’implémentation offre la possibilité d’introduire dans les partitions des points d’interaction permettant à l’interprète de déclencher manuellement certains événements pendant sa performance.

Comme nous ne cherchons pas à synthétiser nous même les sons de la pièce, nous utilisons pour ce faire une application extérieure dédiée (typiquement Max ou Pure Data) avec laquelle le système va communiquer au travers du protocole Open Sound Control selon le schéma de la figure 4.

Ainsi un message OSC est associé à chaque événement, message qui sera envoyé à l’application concernée lors du déclenchement de l’événement. La figure 6 présente les possibilités de configuration de l’utilisation d’OSC par le système. En particulier, on peut voir sur la figure 6(a) la manière de déclarer des nouvelles applications externes au système avec leur caractéristiques : adresse IP et Port d’écoute. La figure 6(b) présente la possibilité de définir les messages OSC associés au début et à la fin d’une boîte. Ces messages seront envoyés à une des applications de la liste d’applications déclarées. Enfin, la figure 6(c) présente la manière d’ajouter un point d’interaction, processus lié

également à OSC dans la mesure où ces points d'interaction seront déclenchés pendant l'exécution par la réception de message OSC par *Iscore*. Sur la partition, la présence d'un point d'interaction est précisé par la coloration de l'événement interactif en rose. On peut voir sur la figure 6(a) que la fenêtre d'ajout d'applications externes permet également de paramétrer le port d'écoute du système.

Comme annoncé précédemment, pour jouer la partition, nous transformons celle-ci en réseau de Pétri avant d'exécuter ce dernier en prenant en compte les déclenchements de l'interprète et en envoyant les messages OSC associés aux événements.

Du côté des applications externes, un routage des messages OSC permet alors de contrôler les déclenchements et relâchements de processus de synthèse par exemple ou la lecture de fichiers comme dans l'exemple de la figure 7.

3.3. Format d'export

Nous avons créé un format XML de sauvegarde des partitions interactives. Pour l'instant elle reste assez succincte et décrit les partitions telles que décrites plus haut avec des boîtes symboliques contenant uniquement des messages OSC associés à leur début et à leur fin. Ce format a pour vocation d'être étendue pour fournir une possibilité d'échange de partitions musicales non conventionnelles et qui échappent à des projet comme MusicXML¹. Notre objectif est également d'utiliser ce format pour permettre des exports dans un format graphique comme SVG² par exemple pour offrir la possibilité d'imprimer les partitions ou encore de les partager sur le net. Un autre format qui paraît également intéressant est SMIL³ qui est un format de description de petits scénarios multimédias avec possibilité de déclencher certains événements de manière interactive. Ce format qui dispose maintenant de plusieurs lecteurs est destiné au partage de ces scénarios sur la toile.

4. PARTICIPATION AU PROJET VIRAGE

Notre approche symbolique des boîtes, nous permet d'envisager la représentation de contenu divers et pas simplement musicaux. Ainsi les processus pilotés par les boîtes peuvent tout à fait synthétiser des images, commander des effets de lumières ou déclencher des effets spéciaux. Cette possibilité nous a amené à considérer des collaborations en dehors du milieu strictement musicales notamment avec le monde du spectacle vivant au travers du projet Virage. Le projet Virage⁴ prévu pour une durée de 2 ans, a pour objectif de développer un outil interactif en direction des régisseurs de spectacle vivant. En effet, le théâtre et les arts du spectacle en général utilisent de plus en plus de contenus multimédia (vidéo, synthèse sonore, effets...), contenus qui s'ajoutent aux contenus artistiques habituellement gérés par les régisseurs (lumière, musique...). Ainsi

¹ <http://www.recordare.com/xml.html>

² <http://www.w3.org/Graphics/SVG/>

³ <http://www.w3.org/AudioVideo/>

⁴ <http://plateforme-virage.org/>

de nombreuses compagnies s'appuient sur l'informatique pour piloter ces éléments pendant les représentations. Or il n'existe pas d'outils réellement pertinents pour écrire des scénarios multimédias organisant l'ensemble des événements et pouvant supporter l'interaction pour assurer une robustesse du système par rapport aux fluctuations temporelles des représentations. On s'aperçoit que l'élaboration d'un tel système est un problème proche de celui auquel nous sommes attelés pour créer *Iscore*. En effet, on peut tout à fait remplacer les processus musicaux associés aux boîtes dans *Iscore* par des processus adéquats pour la lumière, la vidéo... Un régisseur pourrait ainsi organiser temporellement l'ensemble des événements qu'il a à piloter pendant la représentation, synchroniser divers contenus et placer des points d'interaction à des moments stratégiques de la pièce pour pouvoir suivre le jeu des comédiens. Le projet Virage va beaucoup plus loin que la simple adaptation de notre système dans la mesure où une grosse partie de ce projet vise à proposer aux régisseurs un outil permettant de définir les interfaces au travers desquelles il interagira avec le système notamment par l'intervention des sociétés BlueYeti et JazzMutant. Ainsi ce projet devra offrir un outil pour écrire et exécuter des scénarios multimédias interactifs et définir les interfaces avec lesquelles interagir avec ces scénarios.

Notre collaboration dans ce projet va nous permettre de faire aboutir l'implémentation indépendante de *Iscore*.

5. PERSPECTIVES

L'une des voies d'amélioration du système que nous expérimentons est d'ajouter aux relations de Allen la possibilité de définir des contraintes plus fines sur les intervalles de durée. Concrètement, nous souhaitons que le compositeur puisse préciser une propriété pour n'importe quel intervalle de durée séparant deux événements de la pièce. Trois statuts sont possibles :

- rigide : la valeur de la durée de l'intervalle écrite doit être maintenue quoi qu'il arrive
- semi-rigide : la valeur de l'intervalle doit rester dans un ensemble de valeurs $[Val_{min}, Val_{max}]$
- souple : la valeur de l'intervalle peut prendre n'importe quelle valeur dans $[0, \infty]$

Bien entendu, ces contraintes n'ont de sens que par rapport à la présence de points d'interaction susceptibles de modifier les valeurs des intervalles. Après des expérimentations, il est apparu que l'ajout de ces contraintes directement dans le modèle de réseaux de Pétri que nous utilisons n'est pas possible. Nous cherchons donc à ajouter à notre modèle un graphe de contraintes sur lequel appliquer un algorithme de réduction de domaines par propagation de type Indigo [4]. Ce type d'algorithme n'étant pas efficace sur des graphes cycliques (ce qui se produit fréquemment), nous travaillons actuellement à des modifications permettant de prendre en compte ce type de situations.

Une autre amélioration à laquelle nous souhaitons nous attaquer rapidement est d'incorporer progressivement les processus de synthèse dans *Iscore*. Dans un premier temps,

nous allons placer dans les boîtes des courbes de valeurs qui seront lues au déclenchement des boîtes et dont les valeurs seront envoyées via OSC au fur et à mesure que la lecture des boîtes avance, ceci ayant pour but de mieux contrôler les processus de synthèse des applications externes. Nous envisageons également de permettre aux boîtes de recevoir des valeurs et d'enregistrer des courbes lors de l'exécution de la pièce. Par la suite nous envisageons d'embarquer totalement des moteurs de synthèses et de traitements.

Concernant les utilisations possibles de *Iscore*, il y a naturellement la création et l'interprétation de pièces de musique électro-acoustique, une pièce est d'ailleurs actuellement en cours de création par Joseph Laralde, assistant musical au Scrimé⁵.

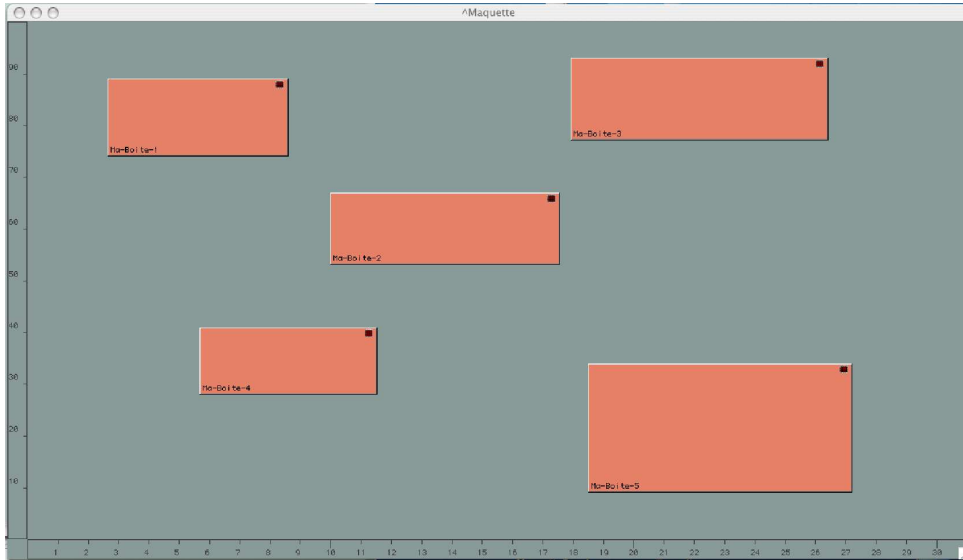
Cependant d'autres utilisations sont envisagées comme par exemple un système de playback qui suivrait le jeu du musicien au travers de points d'interaction. Dans le cadre de cette utilisation, nous cherchons à créer un module de transformation de fichiers Midi et MusicXML en partitions interactives. Ce module devra instancier automatiquement des boîtes et inférer des relations entre elles en fonctions des données du fichier d'entrée. Une dernière utilisation prévue concerne la pédagogie pour permettre à des musiciens débutants de faire l'expérience de l'interprétation de pièce avant de posséder la maîtrise technique pour jouer effectivement la pièce. L'idée est donc de déléguer une partie des notes au système et de conserver un nombre limité de points d'interaction pour contrôler des parties réellement pertinentes pour l'interprétation. D'une manière générale, le système peut permettre d'adapter des pièces aux capacités de jeu des musiciens : débutants, virtuoses, handicapés... Dans ce contexte, le module d'importation de fichiers midi ou musicxml est également central.

6. REFERENCES

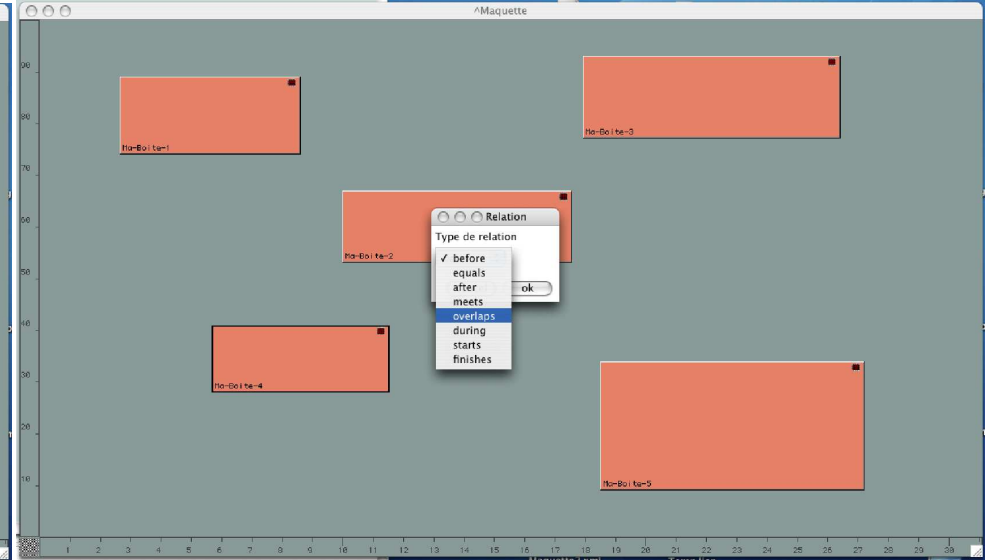
- [1] A. Allombert, G. Assayag, and M. Dessainte-Catherine. A model of interactive scores based on petri nets. In *Pr. of the 4th Sound and Music Computing Conference (2007)*, Lefkada, Greece, July 2007.
- [2] Gérard Assayag, Camilo Rueda, Mikael Laurson, Carlos Agon, and Olivier Delerue. Computer assisted composition at ircam : From patchwork to openmusic. *Computer Music Journal*, 23(3), 1999.
- [3] A. Beurivé. Un logiciel de composition musicale combinant un modèle spectral, des structures hiérarchiques et des contraintes. In *Journées d'Informatique Musicale, JIM 2000*, 2000.
- [4] Alan Borning, Richard Anderson, and Bjorn Freeman-Benson. Indigo : a local propagation algorithm for inequality constraints. In *UIST '96 : Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 129–136, New York, NY, USA, 1996. ACM.

⁵ Studio de Création et de Recherche en Informatique et Musique Électro-acoustique - <http://scrimelabri.fr>

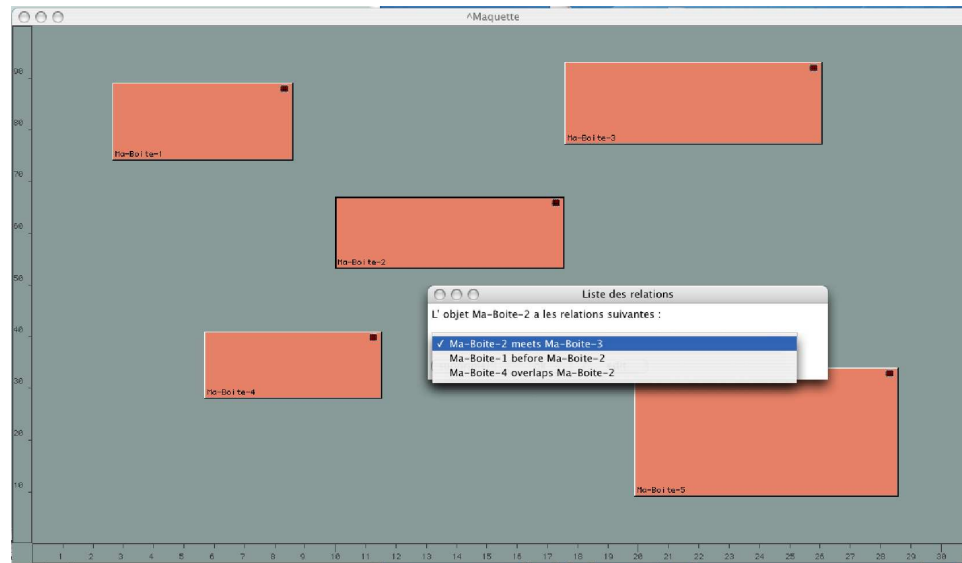
- [5] M. Desainte-Catherine and S. Marchand. Structured additive synthesis : Towards a model of sound timbre and electroacoustic music forms. In *Proc. of the ICMC-99 (International Computer Music Conference)*, Pekin (China), October 1999.
- [6] J. Haury. La grammaire de l'exécution musicale au clavier et le mouvement des touches. In *Manuscrit*.
- [7] F. Pachet and O. Delerue. Musicspace : a constraint based control system for music spatialization. In *Proc. of 1999 International Computer Music Conference*, pages 272–273, 1999.
- [8] C. Schulte and G. Tack. Views and iterators for generic constraint implementations. In *Pr. of the Fifth International Colloquium on Implementation of Constraint and Logic Programming Systems, CI-CLOPS05*, 2005.



(a) Un exemple de partition à partir d'une maquette d'OM

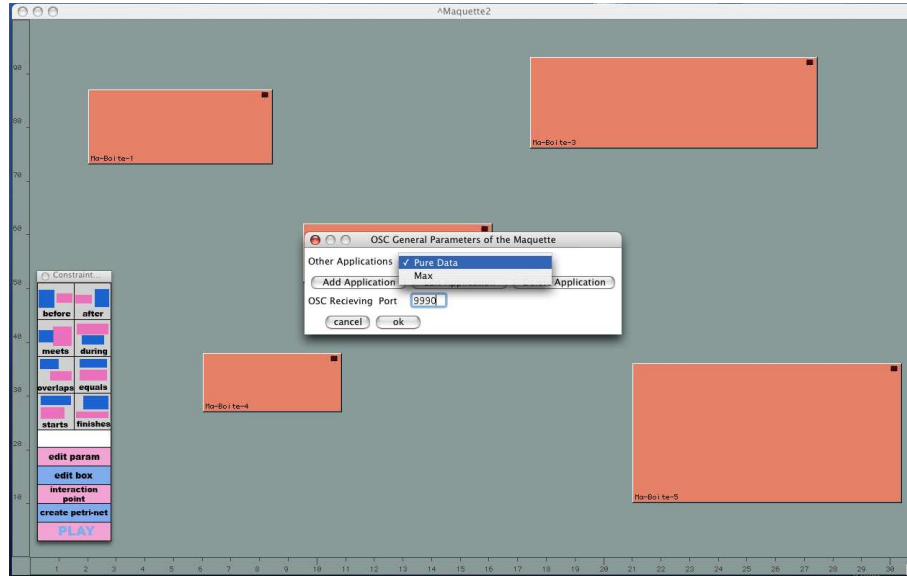


(b) Introduction d'une relation de Allen entre deux boites

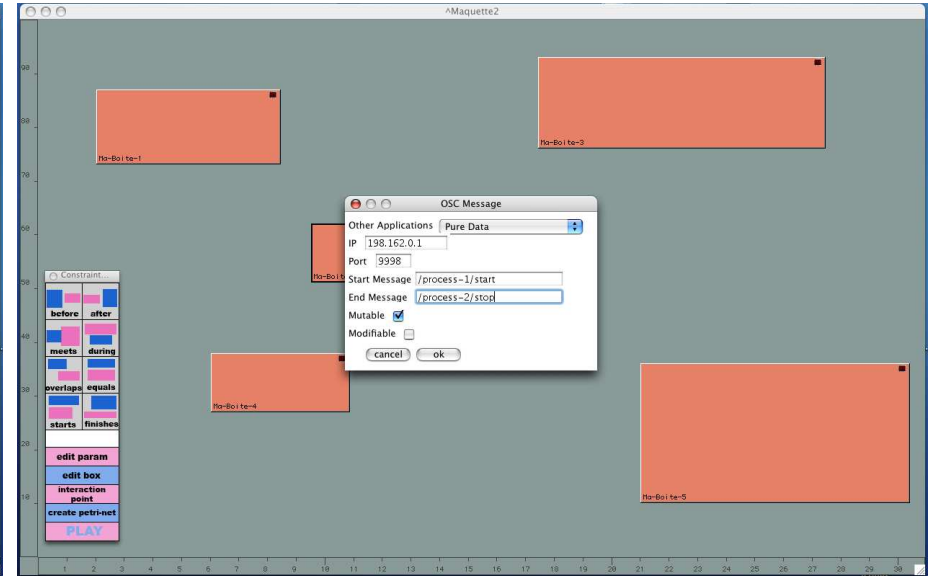


(c) Liste des relations impliquant une boite

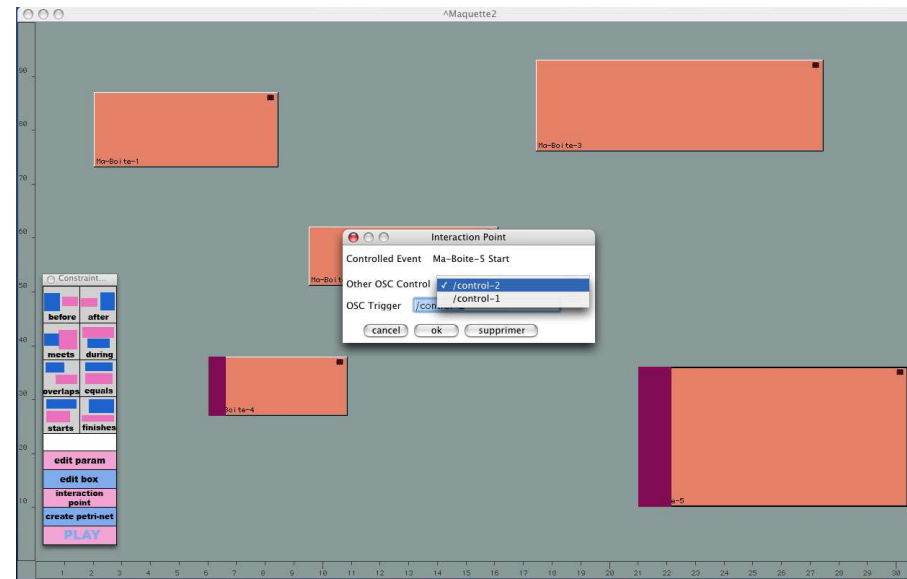
Figure 5. Exemple d'utilisation d'OM pour créer une partition interactive



(a) Paramètre de la partition pour OSC



(b) Définition des paramètres OSC d'une boîte



(c) Ajout de points d'interaction

Figure 6. Paramétrage OSC d'une partition

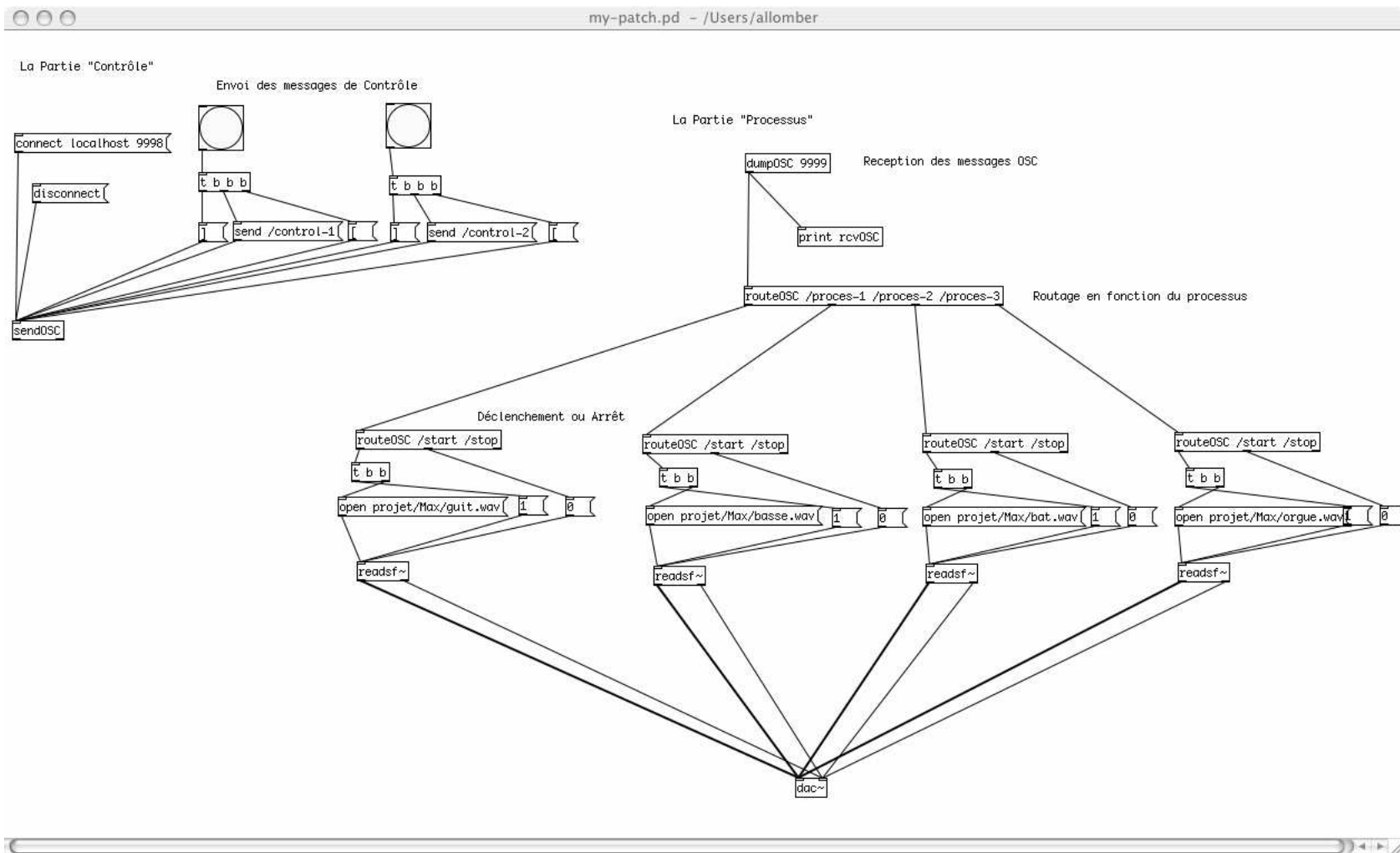


Figure 7. Un exemple de Patch PD utilisé avec Iscore