VIRAGE : UNE REFLEXION PLURIDISCIPLINAIRE AUTOUR DU TEMPS DANS LA CREATION NUMERIQUE

Pascal Baltazar, Antoine Allombert, Raphael Marczak, Jean-Michel Couturier, M. Roy, Anne Sèdes, M. Desainte-Catherine

VIRAGE : UNE REFLEXION PLURIDISCIPLINAIRE AUTOUR DU TEMPS DANS LA CREATION NUMERIQUE

P. Baltazar GMEA
A. Allombert LaBRI
R. Marczak LaBRI
J.M. Couturier Blue Yeti
M. Roy Blue Yeti
A. Sèdes CICM
M. Desainte-Catherine LaBRI

RÉSUMÉ

Nous présentons dans cet article l'évolution du projet Virage dont l'objectif est d'aboutir à une plate-forme autour de l'écriture du temps et de l'interaction dans la régie numérique du spectacle vivant. Ce projet entend répondre à des questions posées par un groupe de travail de l'AFIM. Ses objectifs sont de produire un état de l'art de la pratique de la régie numérique dans le spectacle vivant, puis de réaliser une maquette fonctionnelle basée sur ce premier constat, maquette qui sera mise entre les mains de régisseurs afin d'obtenir des impressions face à un nouveau type d'outils.

1. INTRODUCTION

Le projet Virage prétend aborder des problématiques liées à l'écriture du temps et de l'interaction dans la régie numérique de spectacle vivant et la création d'outils muséographiques. En effet, depuis quelques années, les pratiques des régisseurs et des créateurs ont évolué pour intégrer de plus en plus les outils numériques. Son, image, vidéo, mais aussi avec l'évolution des matériels, lumière ou commandes de machinerie, les différents contenus manipulés par les régisseurs du spectacle vivant peuvent désormais être contrôlés numériquement. Pour faire face à ces évolutions, les praticiens se sont tournés vers des diverses applications souvent dédiées à un type de contenu ou issues d'autres communautés (informatique musicale par exemple). Cependant, les situations de mise en scène de plus en plus complexes rendent difficile l'écriture des contenus dans le temps et de la manière d'interagir avec eux, faute d'outils permettant de mêler la représentation de contenus variés et l'expression des relations temporelles qu'ils entretiennent entre eux. Ces problématiques sont proches de celles rencontrées par les artistes numériques lors de la création d'installations interactives, ou encore lors de l'élaboration d'interfaces pour la muséographie. On retrouve l'écriture dans le temps de contenus variés avec lequel un utilisateur, le visiteur en l'occurrence, va interagir.

Le projet dont nous présentons les avancements ici a pour but d'éclaircir cette question en proposant la maquette d'un séquenceur interactif pour l'écriture du temps et de l'interaction à destination des régisseurs. Cette maquette sera ensuite mise entre les mains des praticiens pour identifier plus précisément leurs besoins dans ce domaine. Compte tenu de la nouveauté que constitue pareil outils vis à vis des utilisateurs, notre méthodologie a mis l'accent sur la pratique des acteurs de terrain afin que leur approche du métier viennent alimenter directement les réflexions théoriques autour du modèle. Des utilisateurs continuellement associés à ces réflexions ont pu suivre l'évolution du modèle pour qu'il reste compatible avec leurs usages.

Nous présentons tout d'abord le contexte dans lequel ce projet est né et les enjeux auxquels il entend répondre. Puis nous développement le modèle théorique sur lequel s'appuie notre prototype avant d'exposer les états d'avancement du développement de cette maquette.

2. CONTEXTE ET ENJEUX

Il convient de signaler en premier lieu que le développement du séquenceur décrit dans cet article a lieu dans le contexte de l'appel à projets Audiovisuel et Multimédia 2007 de l'Agence Nationale de la Recherche, pour la période 2008/2009 dans la catégorie Recherche industrielle. Cette catégorisation implique que l'objectif du projet n'est pas d'aboutir à un produit fini, mais à un cahier des charges pour des développements futurs, en s'appuyant sur une étude des usages réalisée tout au long du projet à partir d'expériences concrètes de mise en oeuvre de prototypes en situation de création artistique. Le champ de recherche du projet en termes de domaine d'activité concerne principalement les métiers créatifs du spectacle vivant, et s'articule autour des notions d'interopérabilité, de modularité, et de l'écriture de comportements de matériaux créatifs (son, image, lumière...) et de leurs interactions autour d'un modèle de "temps souple". La démarche de recherche se veut pragmatique, en confrontant les modèles formels aux pratiques du terrain, par leur implémentation dans des prototypes livrés régulièrement aux gens de l'art et évalués par ceux-ci sous l'oeil de l'équipe du CICM-MSH Paris Nord, chargée de l'étude des usages. En préalable aux premiers développements, et afin d'en définir les directions et priorités, ont déjà été publiés le rapport d'activité du groupe de travail Outils et pratiques du sonore dans le spectacle vivant, missionné

par l'AFIM ([4]), qui a permis l'élaboration du consortium de la plateforme Virage, ainsi que l'étude des usages initiale réalisée au sein du projet ([8]).

La conclusion principale de cette étude des usages a confirmé la nécessité de travailler autour des notions d'interopérabilité et de modularité, en visant la combinaison de nos développements avec les environnements logiciels et matériels existants et correspondant aux habitudes des praticiens, plutôt que l'intégration de cette réalité complexe et multiple dans un illusoire logiciel unique et certainement trop polyvalent pour être viable. Dans le souci de rendre techniquement possible cette interopérabilité, il nous est donc apparu nécessaire de mener un travail de recherche en vue d'un protocole, intitulé Minuit[1], et permettant de relier les environnements avec une implication minimale de l'utilisateur. La société *Stantum*[2] (anciennement *JazzMutant*) conduit cette partie du projet en cherchant à dépasser conjointement les limites des protocoles existants, à savoir MIDI (obsolescence, absence de descripteurs) et OSC (trop grande ouverture, absence de système de découverte des services). Ce souci d'interopérabilité nous a également mené à envisager une certaine indépendance vis-à-vis des media mis en oeuvre dans l'écriture artistique en ne cherchant pas à les contrôler directement, comme c'est le cas dans la plupart des logiciels créatifs, mais en permettant l'écriture de comportements des paramètres de logiciels dédiés.

Cette indépendance à la durée intrinsèque des medias manipulés permet alors d'envisager la mise en oeuvre de la notion de "temps souple", qui est un des enjeux majeurs du spectacle vivant dans la mesure où celui-ci s'élabore en relation à un temps vivant, le temps "du plateau théâtral", le temps humain de l'acteur, du danseur, de l'interprète en général... La relation à ce temps humain est généralement traitée en combinant de façon "artisanale" l'élaboration sur des séquenceurs media dédiés (e.g. Pro-Tools, FinalCut Pro, etc...) de séquences média digées, ensuite combinées et tuilées lors de déclenchements successifs au long du spectacle, comme autant de rendez-vous appelés *cues* par les gens de l'art, et éventuellement enrichis par quelques micro-événements joués en direct sur des dispositifs de type sampler. Ces dispositifs ad hoc ont fait leurs preuves et continueront certainement d'être employés dans le spectacle vivant pendant quelques temps encore, mais il nous a cependant paru pertinent d'explorer la possibilité d'une écriture temporelle à la fois plus souple et plus précise des comportements et interactions des contenus media, grâce au modèle de séquenceur interactif proposé par le LaBRI en étendant aux problématiques du spectacle vivant leurs propositions précédentes BOXES et iScore, comparables à certains égards à une transposition du modèle Gantt à la création musicale.

Comme cela sera décrit en détail ci-dessous, les prototypes du séquenceur interactif Virage permettront de manipuler des objets temporels, pouvant contenir états, comportements et mises en correspondances (ou mappings) de paramètres matérialisés par des messages Minuit envoyés vers des dispositifs numériques de génération et de manipulation de média (développés dans MaxMSP à l'aide des frameworks Tapemovie3 et Jamoma4 pour l'expérimentation dans le cadre du projet). Ces objets temporels seront reliés par des intervalles contraints permettant de définir des relations d'antériorité ou de postériorité (au sens des relations de Allen), correspondant à des durées fixes, souples ou bornées, et permettant ainsi de définir une certaine permanence dans la structure du scénario interactif généré, tout en permettant les modulations en direct lors de l'exécution. Outre l'aspect d'écriture temporelle des comportements des media, le deuxième enjeu fort du séquenceur proposé par la plate-forme Virage concerne l'écriture de l'interaction, soit entre paramètres de dispositifs media hétérogènes (e.g. un paramètre d'analyse sonore agissant sur un paramètre de synthèse vidéo..), soit entre des paramètres d'interfaces gestuelles (que ce soient à l'aide de capteurs sur scène ou de contrôleurs en régie...) et des paramètres de génération ou de manipulation des media. Dans les deux cas, le séquenceur vise la mise en œuvre intuitive et précise de telles transductions entre les différents matériaux créatifs dont la mise en relation dynamique est l'essence du spectacle vivant.

La recherche menée au sein de la plate-forme Virage vise ainsi à repousser les limites de l'existant, notamment concernant le caractère "insulaire" des environnements logiciels ou matériels utilisés par les praticiens du spectacle vivant, ainsi que leurs limitations en termes d'écriture du temps et de l'interaction ([1]). Afin de rester dans une démarche pragmatique, le projet s'appuie cependant sur l'existant, et confrontera ses résultats de manière progressive et incrémentale lors de sessions d'expérimentation et d'évaluation mis en oeuvre par les membres et partenaires de la plate-forme issus de diverses branches de la création artistique, correspondant à autant de situations spécifiques. Ces sessions auront lieu en situation de création artistique, pour être aussi réalistes que possible, quoique hors temps de production afin de ne pas biaiser le travail de recherche par cause de contraintes de résultats artistiques. Elles seront menées par des artistes et compagnies sélectionnés pour la pertinence de leur démarche vis-à-vis des thèmes de recherche du projet, et accompagnés durant les deux ans du projet, afin de garder un suivi de l'évolution. Les axes développés seront en accord avec les orientations des membres les accueillant, à savoir :

[1] La fonctionnalité principale du protocole est la découverte de nom de domaine. Celle-ci permet à une application se connectant à un résau de connaître automatiquement les services disponibles sur ce réseau, ainsi que les messages pour communiquer avec les applications délivrant ces services. Cette fonctionnalité avait été identifiée comme une amélioration sérieuse du protocole OSC [7].

[2] http ://stantum.com

[3] http://tapemovie.org

[4] http://janoma.org

- pour le GMEA : la création sonore pour le spectacle vivant (théâtre, danse) et le concert (situation instrumentale autour des lutheries informatiques...)
- pour didascalie.net : la régie numérique dans le cadre du spectacle vivant, avec un axe fort autour de la vidéo-scénographie (en collaboration avec le LIMSICNRS)

Pour compléter ce panel d'évaluations, plusieurs partenaires tiers se sont joints au projet et mèneront eux aussi des chantiers de recherche et d'expérimentation autour des thèmes qui leur sont propres et qui complèteront ainsi ceux des deux membres de Virage : iMal (Centre des cultures digitales, Bruxelles), GMEM (Centre National de Création Musicale de Marseille), ISTS (Institut de formation aux métiers du spectacle, Avignon), BEK (Centre des arts électroniques de Bergen, Norvège).

3. EVOLUTION DU MODELE THEORIQUE

Dans [3], nous présentions des recherches théoriques ainsi que quelques efforts de développement menés au LaBRI en collaboration avec l'Ircam pour atteindre un système de partitions interactives utilisant des contraintes temporelles Iscore. Nous signalions dans cet article qu'il était question d'adapter notre formalisme à la pratique de la régie numérique du spectacle vivant dans le cadre de la participation du LaBRI au projet Virage. Nous présentons ici les principales modifications que nous avons réalisées. Ces évolutions du modèle font écho d'une part à des perspectives vers lesquelles nous souhaitions nous orienter et d'autre part à des réflexions des utilisateurs associés au projet.

3.1. Retour sur le modèle théorique

Rappelons rapidement quelques points fondamentaux du modèle de partitions interactives pensé pour permettre l'interprétation de pièces musicales numériques. La notion d'interprétation implique qu'un compositeur crée un matériau musical qui sera modifié lors de l'exécution de la pièce par un interprète. Ces possibilités de modification par le musicien sont encadrées par le compositeur qui définit des limites à ces modifications. Nous nous intéressons à des modifications des caractéristiques temporelles de la partition et plus exactement à des décalages sur les débuts et les fins des notes qui les composent. Par conséquent, le cadre fixé par le compositeur concerne l'organisation temporelle de la pièce et il doit être capable de définir une organisation temporelle globale qu'il souhaite voir respectée quoi qu'il arrive. Pour ce faire, il dispose des relations de Allen grâce auxquelles il peut décrire les relations temporelles existant entre les notes. En outre, le compositeur précise explicitement les débuts ou fins de note que le musicien pourra modifier. Ces débuts et fins de note sont appelés les *événements* de la partition. Le compositeur peut rendre un événement *interactif* en y plaçant un point d'interaction ; le musicien pourra alors contrôler directement le déclenchement de l'événement lors de l'exécution de la pièce. Dans le cas contraire, l'événement est dit *statique*. Notons que pour affiner l'écriture de l'organisation temporelles de sa pièce, le compositeur dispose également de relations temporelles quantitatives : il peut préciser si un intervalle de temps séparant deux événements doit rester constant, peut être modifier dans une certaine mesure ou prendre n'importe quelle valeur. La version écrite de la partition ne représente donc plus qu'une possibilité d'interprétation parmi de nombreuses autres. Naturellement ; le système maintient les contraintes temporelles imposées par le compositeur : si au cours de l'exécution le musicien décale le déclenchement d'un événement interactif, le système va recalculer les dates des événements non encore déclenchés pour qu'elles respectent la date effective de l'événement interactif et les contraintes temporelles.

Précisons enfin que pour la phase d'exécution d'une partition, nous la compilons en une représentation interprétable par une machine abstraite capable d'exécuter le contenu musical de la partition tout en réagissant aux déclenchements des événements interactifs. Cette compilation implique une transformation en réseaux de Pétri ; pour plus de détail, le lecteur pourra se référer à [2].

Dans le cadre de Virage, ce modèle théorique a été repris dans la mesure où les notes des partitions pouvaient également représenter des processus manipulant des contenus autres que musicaux (vidéo, lumières...) et en particulier des processus qu'on retrouve dans la régie numérique. Pour s'adapter à cette nouvelle utilisation, le modèle a connu une modification de vocabulaire. Le terme "partition" a laissé la place à celui de "conduite" tandis que les notes devenaient des "objets temporels".

3.2. Les processus

Dans la version précédente de *Iscore*, le contenu des notes restaient en dehors de notre champs d'investigation. Sans nous atteler à résoudre totalement la question de la représentation des contenus des objets temporels, nous avons introduits la notion de *processus*. Un processus correspond à une opération algorithmique déterministe et non-conditionnelle qui possède éventuellement des *entrées* pour accepter des arguments. De plus un processus exhibe des *étapes de calcul* qui représentent des moments caractérisés de son algorithme. Les étapes les plus naturelles d'un processus sont son début et sa fin, cependant dans le cas d'un processus ponctuel la fin du processus n'existe pas. En outre, des étapes intermédiaires peuvent être spécifiées, un exemple simple étant un processus de lecture d'une courbe de valeurs dont on va caractériser le passage par un extremum. Le caractère déterministe et non-conditionnel des algorithmes associés nous permet d'affirmer que le nombre d'étapes d'un processus est

constant à chaque exécution et qu'elles se déroulent toujours dans le même ordre. Un objet temporel représente une exécution dans le temps du processus qui lui est associé.

3.3. Les points de contrôle

En lien avec les étapes de calcul des processus, nous avons introduit des points intermédiaires dans le déroulement des objets temporels. Ces points qui s'ajoutent aux débuts et aux fins d'objets, ont naturellement pour but de représenter les moments de déclenchement des étapes de calcul des processus associés. Ils sont appelés *points de contrôle*. Formellement, soit OT un objet temporel de durée d, un ensemble de points de contrôle $\mathcal{P}$ datés lui est associé. On a :

- si $\mathcal{P}$ est réduit à $\{(start(OT),0)\}$ alors OT est ponctuel et d=0
- sinon $\mathcal{P} = \{(pc_1,0),...,(pc_i,d_i),...,(pc_n,d)\}$ avec $pc_1 = start(OT)$ et $pc_n = end(OT)$

Les points de contrôle pci sont datés relativement à la date de début de l'objet. On voit bien ici qu'un objet temporel est la représentation en temps d'un processus abstrait puisqu'il permet de dater le déclenchement des étapes de calcul du processus.

Les points d'interaction peuvent être définis sur n'importe quel point de contrôle.

3.4. Les relations temporelles

Dans notre modèle précédent, on pouvait spécifier une seule relation de Allen entre deux objets ce qui réduisait les possibilités par rapport au formalisme de Allen qui utilise les opérateurs *et* et *ou* pour construire une grammaire à partir des relations de base. En outre, ces relations de base ne semblait pas convenir aux utilisateurs du spectacle vivant. L'introduction des points de contrôle nous a permit de répondre à ce double problème en remplaçant les relations de Allen originelles par des relations temporelles définies entre deux points de contrôle et qui spécifie les caractéristiques de l'intervalle de temps entre ces points. Plus précisément, on trouve deux cas de relations temporelles : **pre** et **post**.

Soient deux points contrôle pc_1 et pc_2, une relation $(pre,pc1,pc2,\Delta min,\Delta max)$ impose la contrainte :

$\Delta_{min} \leq date(pc_2) - date(pc_1) \leq \Delta_{max}$

Les valeurs des Δ sont non-nulles et éventuellement infinies. Une relation $(post,pc_1,pc_2,\Delta_{min},\Delta_{max})$ est équivalente à une relation $(pre,pc_2,pc_1,\Delta_{min},\Delta_{max})$. Ces nouvelles relations permettent de préciser dans le même temps des contraintes quantitatives sur les intervalles de temps. On peut reconstruire les relations de Allen originelles en combinant des relations **pre** et **post** sur les débuts et fins d'objet. L'écriture du temps devient donc la spécification d'intervalle séparant des événements de la conduite.

La figure 1 présente un exemple de conduite interactive. On y trouve deux objets dont les processus associés sont des lectures de courbes. Le point de contrôle intermédiaire de l'objet *son* est rendu interactif par l'ajout d'un point d'interaction et une relation temporelle **pre** est définie entre les deux débuts.

3.5. Machine ECO

La machine ECO est une machine abstraite permettant l'exécution des partitions [6]. Elle s'exécute en interprétant un environnement contenant toutes les informations nécessaires au déroulement de la conduite ainsi qu'à l'interaction. Pour créer l'environnement, la conduite est compilée, les informations temporelles sont représentée par un réseau de Pétri associé à un solveur de contraintes.

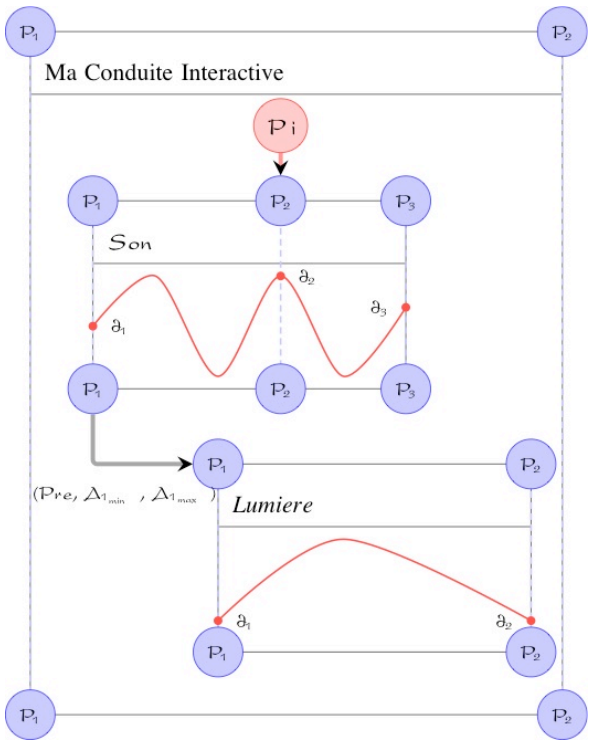


Figure 1. Un exemple de conduite interactive

Nous avons ajouté au formalisme de la machine une partie concernant les processus. L'ajout des points intermédiaires modifie quelque peu l'algorithme de compilation des partitions en réseaux. Cependant chaque point est représenté par une transition, les places d'attente du réseau ainsi que les arcs reliant les éléments traduisent les contraintes temporelles de la partition. Le CSP permet de réagir aux modifications dues à l'interaction de l'utilisateur. Au cours de l'exécution, lorsqu'une transition est franchie, le réseaux envoie un message au processus concerné pour qu'il déclenche l'étape de calcul associée à la transition. La figure 2 présente cette machine.

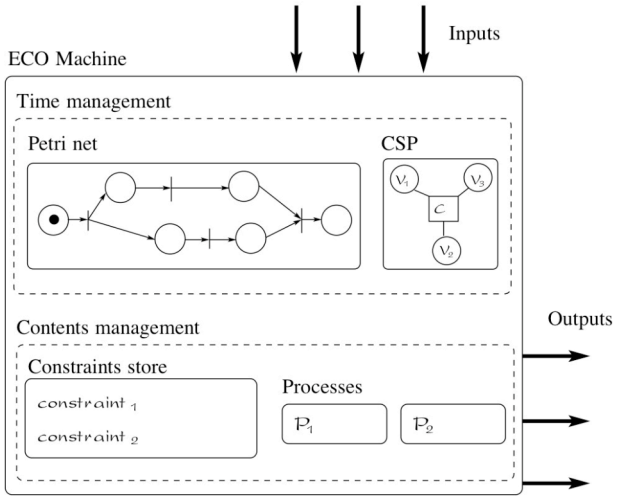


Figure 2. La machine ECO

4. IMPLEMENTATION

Dans cette partie, nous allons présenter l'implémentation du moteur, en décrivant l'architecture, les fonctionnalités, son état actuel et les évolutions prévues sur le moyen terme. Cette architecture correspond à l'implémentation du formalisme de conduites interactives précédemment décrit. Par moteur, nous parlons en réalité de la réunion de deux moteurs, correspondant à deux phases essentielles de la création d'un spectacle vivant : un moteur d'édition pour la composition, et un moteur d'exécution pour la performance. Le moteur d'édition permet de créer et manipuler les conduites tandis que le moteur d'exécution permet l'interprétation de la conduite.

4.1. Environnement

Le moteur est implémenté en C++ sous Mac OSX ; sa licence libre rend le code source entièrement disponible et observable, permettant entre autre l'ajout de fonctionnalités spécifiques. Le développement au sein de VIRAGE a été grandement facilité par les développements précédents d'Iscore par Antoine Allombert et Bruno Valèze :

- Antoine Allombert a développé le moteur d'exécution en Common LISP dans OpenMusic, lors du déroulement de sa thèse. Une grande partie du code présent dans le moteur d'exécution de VIRAGE provient d'un portage de son code.
- Bruno Valèze a développé pour BOXES un moteur d'édition en C++, basé sur le solveur de contraintes GeCode. Ce moteur est malheureusement très dépendant de son interface graphique sous QT. De plus, il utilise encore les relations de Allen, maintenant remplacées par des relations d'antériorité et de postériorité. Le moteur d'édition présent dans VIRAGE s'inspire fortement de ce développement, mais n'a pas pu l'utiliser en l'état.

4.2. Objectifs

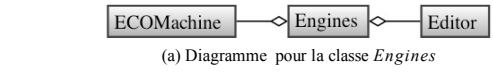
Les objectifs de l'implémentation de ce moteur sont multiples :

- Proposer des fonctionnalités d'édition et d'exécution qui seront testées par les utilisateurs associées au projet.
- Implémenter la totalité du formalisme de conduites interactives.
- Rendre totalement indépendant le moteur de l'interface graphique
- Réussir à le rendre multiplateforme.

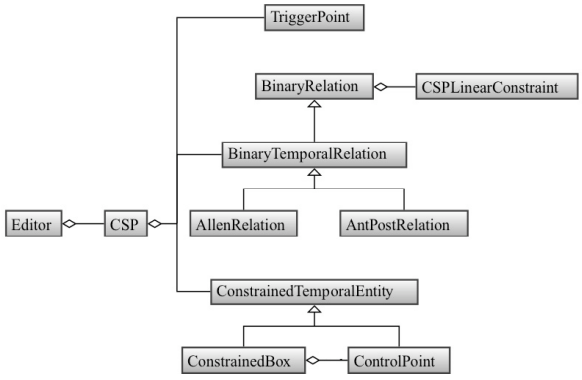
4.3. Architecture

4.3.1. Classe Engines

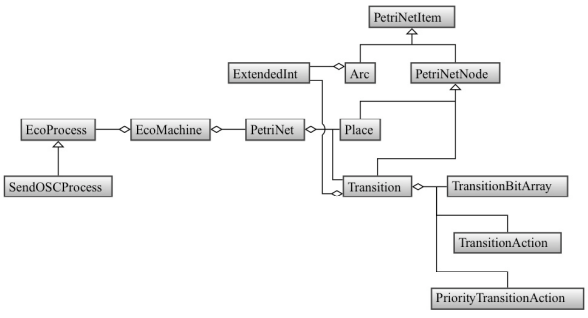
Engines est une classe permettant de rendre plus facile l'utilisation du moteur ; par exemple pour le développeur d'une interface graphique. En effet, pour éviter aux utilisateurs du moteur de rentrer dans les détails d'implémentation, cette classe centralise toutes les fonctions nécessaires à la composition et à l'exécution. Ainsi, l'utilisateur n'a pas à se demander ce qui a trait au moteur d'édition, ou ce qui a trait au moteur d'exécution. Il suffit donc à l'utilisateur d'instancier un objet Engines pour disposer de toutes les fonctionnalités implémentées. Nous verrons ces fonctionnalités dans le chapitre suivant.



(a) Diagramme pour la classe *Engines*



(b) Diagramme pour la classe *Editor*



(c) Diagramme pour la classe *ECOMachine*

Figure 3. Diagrammes de Classe

4.3.2. *Classe Editor*

Editor est une classe permettant la mise en place de la composition avec des objets temporels contraints (ou non) entre eux. Chaque boite créée contient automatiquement deux points de contrôle : un au début, et un à la fin. Ces points contiennent une variable représentant leur date de début, et cette variable est ajoutée au solveur de contraintes GeCode. A chaque modification, par exemple l'ajout d'une nouvelle relation, le solveur de GeCode met à jour toutes ses variables pour fournir une solution respectant les contraintes. Cette classe permet également l'ajout d'interaction avec les points de déclenchement. Chaque objet temporel créé est manipulable grâce à un identifiant unique.

4.3.3. *Classe ECOMachine*

ECOMachine est une classe permettant l'exécution d'une conduite. Elle implémente le modèle de machine ECO présenté plus haut, elle contient donc un interpréteur de réseau de Pétri et un gestionnaire de processus. Les parties concernant la résolutions de contraintes en temps réel ne sont pas encore intégrée. Les fonctionnalités des processus ne sont pas limitées à celles implémentées pour l'instant, il leur suffit de pouvoir comprendre l'information fournit par le réseau de Petri à chaque franchissement de transition. Tout processus héritant de la classe ECOProcess répond à cet exigence. Pour le moment, un seul type de processus est proposé. Il permet l'envoie d'un message OSC au début et à la fin d'une boîte. Mais dans un futur proche, il existera d'autres processus, comme l'envoie de courbe, le mapping, la captation scénique, ...

4.4. Fonctionnalités

Nous présentons les différentes fonctionnalités utilisateurs du moteur.

– Edition

- addBox : ajoute une boite, et fournit à l'utilisateur son identifiant unique.
- removeBox : supprime une boite.
- addAntPostRelation : ajoute une relation entre deux points de contrôle, et fournit à l'utilisateur son identifiant unique, ainsi que la liste des boites dont la position est modifiée pour respecter les contraintes.
- removeTemporalRelation : supprime une relation.
- performMoving : déplace une boite, et fournit à l'utilisateur la liste des boites dont la position est modifiée pour respecter les contraintes.
- getBeginValue : fournit à l'utilisateur la date de début d'une boite.
- getEndValue : fournit à l'utilisateur la date de fin d'une boite.
- nbControlPoint : fournit à l'utilisateur le nombre total de points de contrôle contenus dans la boite.
- setControlPointMessageToSend : spécifie unmessage à envoyer lors du franchissement d'un point de contrôle.
- removeControlPointMessageToSend : enlève le message précédemment spécifié.
- addTriggerPoint : ajoute un poit de déclenchement, et fournit à l'utilisateur son identifiant unique.
- removeTriggerPoint : supprime un point de déclenchement.
- setTriggerPointRelatedControlPoint : relie un point de déclenchement avec un point de contrôle.
- removeTriggerPointRelatedControlPoint : supprime le lien entre un point de déclenchement et un point de contrôle.
- setTriggerPointMessage : spécifie le message déclenchant le point de déclenchement.
- getTriggerPointMessage : fournit le message spécifié précédemment.

– Passage de l'édition à l'exécution

- compile : créé le réseau de Petri du moteur d'exécution à partir du moteur d'édition.

– Exécution

- setIpAddress : indique l'adresse de l'application avec laquelle le processus va communiquer.
- getIpAddress : fournit l'adresse précédemment spécifiée.
- setPort : indique le port de l'application avec laquelle le processus va communiquer.
- getPort : fournit le port précédemment spécifié.
- setTriggerPort : spécifie le port d'écoute pour les messages de déclenchement.
- getTriggerPort : fournit le port précédemment spécifié.
- run : lance l'exécution de la conduite.
- stop : arrête l'exécution de la conduite.
- isRunning : indique si une exécution de la conduite est en cours.

– Une fonctionnalité de sauvegarde et de chargement à partir d'un format XML dédié est également implémenté.

4.5. Evolutions

Le moteur en est pour le moment à la moitié de son développement au sein projet VIRAGE. Les premiers retours d'utilisation vont bientôt nous parvenir, et nous pourrons alors rajouter des fonctionnalités essentielles pour les créateurs de spectacles vivants. Voici une liste non exhaustive des prochains développements, sachant que cette liste sera mise à jour après les premiers tests en situation :

- Ajout de la hiérarchie : boites dans des boites.
- Ajout de plusieurs point de contrôle au milieu des boites ; ne pas se limiter à 2 points.
- Ajout des processus de contrôle continu : envoi de courbe.
- Ajout du mapping.
- Ajout d'une fonction d'accélération du déroulement lors de l'exécution.
- Ajout d'une fonction permettant d'aller à un point quelconque de la composition.

5. CHOIX ET CONCEPTION DE L'INTERFACE GRAPHIQUE

5.1. Ecriture du temps souple - gestion de scénario interactif

L'enjeu central autour de la question des interfaces Homme- Machine pour le séquenceur VIRAGE est de permettre l'écriture du temps souple par l'intermédiaire d'une interface graphique adaptée. Celle-ci doit permettre à des utilisateurs non-spécialistes de la programmation informatique de créer des scénarios interactifs complexes. Le travail de recherche mené autour de la question des interfaces a pour but de rendre transparent à l'utilisateur le formalisme sous-jacent à la création des scénarios.

Cette interface sera compatible avec les périphériques d'entrée usuels (clavier, souris) mais également avec d'autres types d'interfaces de contrôle, comme celles qui seront développées par le LIMSI, notamment pour l'édition des données contenues dans le scénario interactif. On s'appuiera sur le protocole de communication MINUIT pour permettre la réception et l'envoie de données ; l'interface graphique permettra également de paramétrer l'envoi et la réception de données minuit.

En mode écriture, le scénario interactif sera édité graphiquement, par l'assemblage d'éléments graphiques correspondant à des objets temporels et des intervalles permettant de créer des contraintes temporelles entre ces objets. Ces objets temporels pourront être associés à des paramètres de contrôle de modules multimédia (générateurs de sons, de vidéo, lumières,. . .) ou des regroupements de ces paramètres. Des points de déclenchements pourront être placés sur l'interface et reliés aux objets temporels ; en mode exécution, ils permettront de déclencher l'exécution d'un objet temporel ou de terminer son exécution. Le basculement en mode exécution se fera par l'intermédiaire d'une commande d'exécution (bouton "play") et l'interface graphique permettra de visualiser le déroulement du temps et l'exécution des actions.

De très nombreux paramètres devront être manipulés en cours d'édition ou en cours de jeu, ou encore restitués pour information : comment alors ne pas surcharger l'interface et la rendre utilisable par l'utilisateur ? Une démarche de conception centrée utilisateur doit donc être adoptée.

5.2. Etat de l'art des IHM et étude des usages

La première étape a été la constitution d'un état de l'art des IHM des logiciels existants, tels que séquenceurs audio, logiciels de montage vidéo, logiciels d'animation, outils de gestion du temps . . . Cette étude visait à explorer et commenter les différentes solutions d'interfaces mises en œuvre pour effectuer des tâches précises selon différents critères d'analyses :

- partie séquenceur : positionnement de données/infos dans le temps, navigation dans le temps, navigation dans les données, actions sur le temps, actions sur les données
- partie interactive : Ajout de contraintes temporelles, création de scénarios interactifs, mapping et routage
- critères ergonomiques : visualisation / représentation des données, accès aux fonctions, aspect visuel, types d'interactions mises en jeu, niveau d'expertise demandé, degré d'interaction [5], personnalisation possible de l'interface graphique

L'étude des usages menée par le CICM et destinée à restituer la façon dont travaillent les praticiens, nous a apporté un cadre de contraintes propres aux utilisateurs concernés par le séquenceur VIRAGE et nous a permis d'orienter nos choix d'interfaces pour la conception des premières maquette d'interface 5 [5]. Ces deux études ont permis d'établir un cahier des charges fonctionnel de l'interface. Ci-dessous nous en indiquons les principaux blocs de fonctions :

- Visualisation des événements temporels et navigation dans le scénario interactif
- Construction du scénario : insertion d'objets temporels, d'intervalles, de points de déclenchement et manipulation de ceux-ci, possibilités d'imbrications d'éléments dans un objet temporel
- Edition des contenus de chaque blocs
- Gestion des données : courbes
- Mapping/routage : gestion des entrées sorties MINUIT, création de mapping entrées / sorties
- Contrôle du scénario en mode exécution
- Gestion de bibliothèques de scénarios facilitant l'édition
- Déport des interfaces des logiciels contrôlés par le séquenceur
- Gestion de configurations de paramètres minuit : sélection d'ensembles de paramètres, captures d'états, association aux objets temporels
- Fonctions générales du logiciel

5.3. Maquettes de principe et maquettes fonctionnelles d'interface

Les maquettes papier proposent ainsi un premier principe fonctionnel d'interface. Ces maquettes ont été présentées et évaluées par les praticiens et chercheurs du projet. Durant la première partie du projet, les maquettes papier ont donc évolué jusqu'à aboutir à un modèle permettant de répondre aux différents cas pratiques pouvant être rencontrés sur le terrain. Les figures 4 et 5 présentent des versions successives de ces maquettes papier.

[5] http://www.plateforme-virage.org/publication-etude-des-usages.html

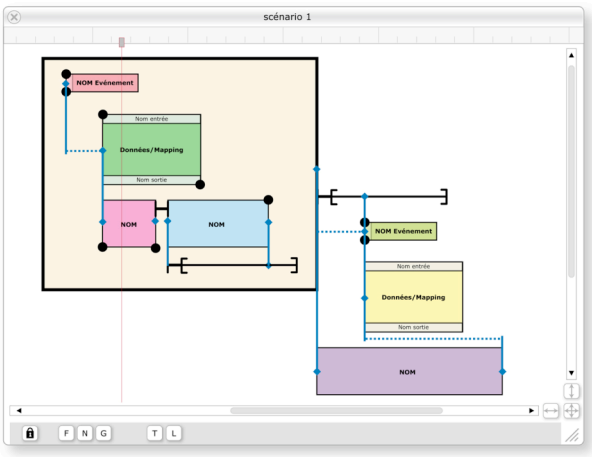


Figure 4. Version 1 de la maquette de principe de l'interface du séquenceur VIRAGE : fenêtre de visualisation du scénario interactif

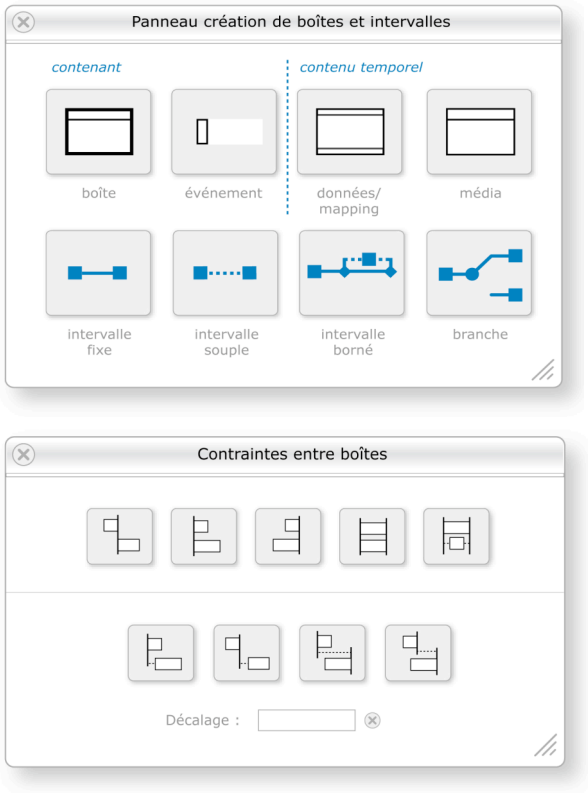


Figure 5. Version 3 de la maquette de principe : panneau de création de boîtes et intervalles et panneau contraintes entre boîtes

Les maquettes papier modélisent l'ensemble des fonctions déterminées dans le cahier des charges et proposent une première représentation d'interface dans laquelle les fonctions sont représentées et positionnées, sans critères graphiques. Pour évaluer ces maquettes, nous les avons mises en situation avec des cas concrets apportés par les praticiens et observés dans quelle mesure elles permettaient ou non de construire un scénario interactif.

Comme déjà évoqué dans la partie concernant le formalisme, nous avons préféré parler d'intervalles contraints plutôt que de relations de Allen. Du point de vue utilisateur, la notion de relations de Allen, même si elle permettent de bien décrire formellement les différentes contraintes entre objets temporels d'un scénario interactif, n'est pas facilement accessible pour l'utilisateur et ne correspond pas toujours à la logique pratique de création d'un scénario et de créations de contraintes entre événements et blocs temporels.

Après la présentation des premières maquette papier, nous avons démarré le développement d'une maquette fonctionnelle du séquenceur interactif. Cette maquette fonctionnelle permettra de tester en situation réelle le principe du séquenceur VIRAGE. La figure 6 présente une capture d'écran de la version actuelle de la maquette fonctionnelle.

L'environnement de développement retenu pour la création de l'interface graphique est *JUCE*[6] , notamment utilisé pour le développement de l'interface graphique de l'application MAX5. Cette maquette correspond à la partie interface graphique développée par Blue Yeti associée au moteur développée par le LaBRI. La première version de la maquette est mise à disposition en février 2009 aux participants du projet VIRAGE. Cette version doit permettre de placer des objets temporels, définir leur comportement, les contraindre par l'ajout d'intervalles fixes, souples ou bornés, définir des points de déclenchement pour démarrer ou terminer un objet temporel, définir des messages Minuit qui pourront être envoyés en début et fin d'exécution d'objets temporels et exécuter le scénario.

La version suivante intégrera :

- Les fonctionnalités complètes de configuration d'échanges de données MINUIT (choix des paramètres à contrôler ou à recevoir), création de configuration de paramètres (presets, capture d'états), mémorisation de configuration de paramètres
- Édition de courbes, automation (enregistrement de données entrantes sous forme de courbe)
- Gestion de mapping entre entrées et sorties

Cette première version sera évaluée par le CICM lors de différents tests en contexte de production artistique auprès des divers partenaires du projet Virage (GMEA, Di-dascalie.net, ISTS Avignon, MSH Paris Nord). Les retours des utilisateurs seront analysés et transmis à l'ensemble de l'équipe. Le protocole d'évaluation sera précisé par le CICM lors de la livraison de la maquette.

6. CONCLUSION

Nous avons présenté un bilan à mi-parcours du projet *Virage* visant à la création d'une plate-forme autour de l'écriture du temps et de l'interaction dans la régie numérique dans le spectacle vivant. Déjà, plusieurs objectifs du projet ont été atteints : un état de l'art de la

[6] http ://www.rawmaterialsoftware.com/juce/

pratique actuelle des régisseurs a pu être dressé par le CICM. De ce travail, ainsi que la réflexion avec les régisseurs associés au projet, a pu naître une description formelle de ce que devrait être une conduite interactive et des mécanismes permettant de la manipuler. Partant de là, l'implémentation de la maquette du séquenceur a pu être entamée et les premiers test seront effectués très prochainement. Outre la création de la maquette, le projet Virage aura permis d'entamer la réflexion autour de problématiques complexes en maintenant l'interdisciplinarité entre des communautés diverses. Nous estimons que la méthodologie ayant permis de maintenir ce dialogue est un apport du projet.

7. REFERENCES

[1] Outils et pratiques du sonore dans le spectacle vivant, rapport du groupe de travail interdisciplinaire de l'afim. Technical report, Association Française d'Informatique Musicale. www.afimasso. org/IMG/pdf/GT_son_spect_viv.pdf.

[2] A. Allombert, G. Assayag, and M. Desainte-Catherine. A system of interactive scores based on petri nets. In *Pr. of the 4th Sound and Music Computing Conference (2007), Lefkada, Greece,* July 2007.

[3] A. Allombert, G. Assayag, and M. Desainte-Catherine. De boxes core : vers une ecriture de l'interaction. In *Pr. of the 13th Journées d'InformatiqueMusicale (2008), Albi, France,* Mars 2008.

[4] P. Baltazar and G. Gagneré. Outils et pratique du sonore dans le spectacle vivant. In *Actes des Journées d'Informatiques Musicales (JIM07), Lyon,* France, pages 153–162, Avril 2007.

[5] M. Beaudouin-Lafon. Instrumental interaction : an interaction model for designing post-wimp user interfaces. In CHI Letters : *Proc. of the ACM Human Factors in Computing Systems (CHI 2000)*, La Haye, Pays-Bas, volume 2, pages 446–453. ACM Press, Avril 2000.

[6] M. Desainte-Catherine and A. Allombert. Interactive scores : Amodel for specifying temporal relations between interactive and static events. *Journal of New Music Research*, 34(4), December 2005.

[7] T. Place, T.Lossius, A. Refsum Jensenius, N.Peters, and Pascal Baltazare. Addressing classes by differentiating values and properties in osc. In *Proc. of the 8th International Conference New Interfaces for Musical Expression, NIME08, Genova, Italy,* June 2008.

[8] A. Santini, A. Sédès, and B. Simon. Etude des usages, des outils, et des besoins dans les métiers de la régie numérique dans le spectacle vivant. etat de art. Technical report, CICM, MSH Paris Nord, Novembre 2008. http ://www.plateforme-virage.org/publicationetude- des-usages.html.

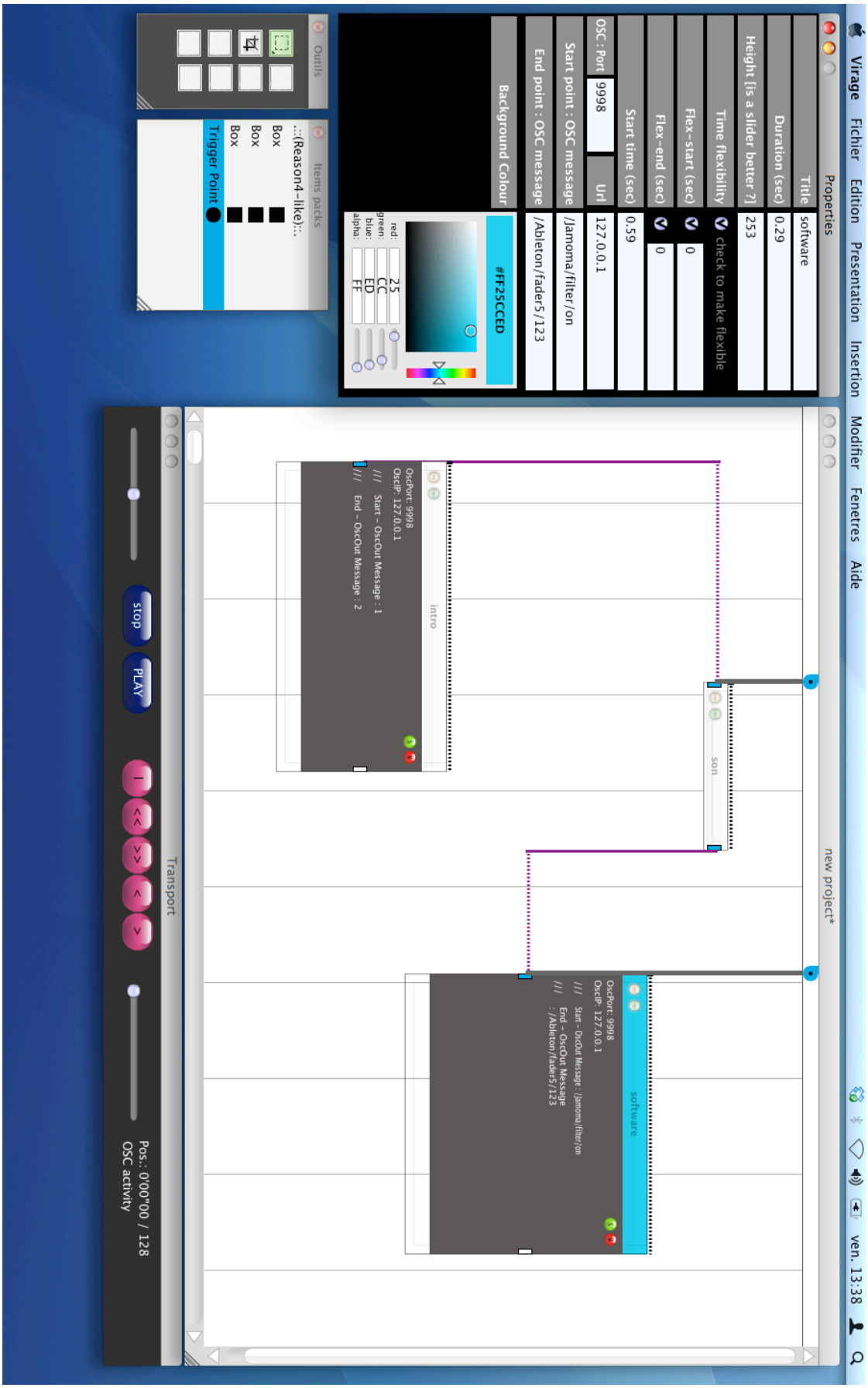


Figure 6. Une capture d'écran de la maquette fonctionnelle