TABLE DES MATIÈRES

Page

LISTE DES FIGURES

Page

INTRODUCTION

> Musical generation is a challenge for analyzing how human mind enjoys melodic sequences and harmony. (Situngkir, 2010)

La composition musicale est l'action de créer une œuvre musicale par un processus de création dont le compositeur est l'acteur principal. L'envie d'exprimer une idée musicale précise peut être un élément déclencheur du processus, mais la motivation peut découler du besoin d'exprimer une idée non musicale telle qu'un sentiment ou un état d'esprit. Les formes musicales sont diverses en occident allant de la musique tonale (musique classique, musique rock par exemple) à la musique concrête. En fonction des styles et des compositeurs, les processus diffèrent grandement. Ils se subdivisent aussi en sous-processus. Par exemple, l'écriture d'une mélodie, l'arrangement d'une pièce, le travail des sons et du timbre des instruments peuvent constituer toute ou une partie du processus de composition.

Le processus de composition représente donc un ensemble d'activités qui sont choisies par le compositeur, pour composer. Certaines de ces activités peuvent dépendre d'outils que le compositeur fait le choix d'utiliser (les règles de solfège sont des outils, tout comme des logiciels d'aides à la composition). Ces choix ont un impact direct sur le résultat.

Dans le contexte de ce travail, la création de mélodies est la partie du processus de composition qui a succité le principal intérêt. Elle ne fait pas nécessairement partie de tous les processus de composition et est absente par exemple en musique concrète : il n'y a pas de mélodie en musique concrète. Mais dans la majorité des styles constituants la musique occidentale (tels que le rock, le jazz, la musique classique et la musique électronique) les compositeurs créent des mélodies et les utilisent dans leurs compositions. C'est un processus dont les activités sont difficiles à définir pour un compositeur en particulier et impossibles à définir pour tous, puisque chaque compositeur applique des processus différents. Mais il existe des similitudes, il y a des activités qui reviennent très souvent, dépendamment des styles de musique, comme le choix des progressions d'accords et des gammes (ce qui définie l'harmonie).

Afin de soutenir les processus de compositions, il existe les logiciels informatiques d'aide à la composition. Chacune de ces solutions est pensée pour soutenir certains types de processus de création et donc certains styles de musique et vont répondre aux besoins de certains compositeurs. Certains logiciels offrent de soutenir des activités du processus (l'écriture d'une partition en notation standard dans un logiciel peut remplacer l'écriture sur papier) ou bien proposent de nouvelles façons de créer à l'aide de nouveaux outils (superposition de boucles midi ou audio avec un séquenceur tels qu'Ableton Live Ableton (2013), difficilement réalisable sans un séquenceur).

La différence entre la composition aidée par ordinateur et la composition automatique est exprimée dans Anders *et al.* (2003). La première a pour vocation d'assister le compositeur humain dans le processus de création alors que la seconde automatise le processus en excluant l'humain de l'exécution de ses différentes tâches.

Les systèmes de composition automatique ne sont pas très répandus et la plupart sont des prototypes de recherche tels que le prototype Hyperscore de Morwaread *et al.* (2007) ou les générateurs de mélodies « Melody Generator » de Povel (2010) et APOPCALEAPS de Sneyers et De Schreye (2010). Mais certaines applications commerciales existent telles que le générateur d'accompagnement SongSmith (originellement le prototype MySong de Simon *et al.* (2008)) ou bien encore le générateur de mélodies Noaktil (Intermorphic (2013)). Ce dernier génère en temps réel et réserve donc plutôt son utilisation à de performances musicales.

Après avoir testé certaines des applications ci-dessus, dont le Melody Generator de Povel (2010), et après s'être bien documenté au sujet des algorithmes de génération, il nous a semblé qu'il existe des algorithmes de génération assez performants pour générer des mélodies pouvant satisfaire l'utilisateur. Malheureusement les travaux précités se concentrent sur le développement des algorithmes et les interfaces utilisateurs existent seulement dans le but de montrer les résultats que ces algorithmes sont susceptibles de produire. À l'aide de ces interfaces utilisateurs, le choix des paramètres de génération nous a paru souvent complexe et limité : navigation compliquée dans les menus de Noaktil ; impossibilité de faire varier les paramètres dans le temps (pour le Melody Generator, par exemple) ; impossibilité d'accéder

à un historique des paramètres de génération précédemment utilisés (pour tous) ; possibilités d'éditions des mélodies générées inexistantes ou très limitées (le Melody Generator permet seulement de supprimer ou de régénérer un passage). Ces limitations réservent l'utilisation des algorithmes du Melody Generator et de APOPCALEAPS à l'expérimentation, et réservent l'usage de la solution commerciale Noatil à un public d'utilisateurs expérimentés. De plus, la réutilisation des mélodies générées comme matière pour de nouvelles générations n'est pas envisagée. Noaktil permet de générer plusieurs voix audio en même temps, mais ne permet pas de les éditer en midi. Les autres prototypes ne le permettent pas non plus.

Les applications de composition aidée par ordinateur et de composition automatique se situent sur un large spectre de niveaux accessibilités adaptées aux différents groupes d'utilisateurs, allant du musicien débutant au musicien confirmé, du néophyte en technologies à l'expert en informatique. Certains ciblent une population très large d'utilisateurs musiciens comme le séquenceur Ableton Live (Ableton (2013)) et le générateur d'accompagnement Band-in-a-Box de PGMusic (Gannon (1990)). D'autres ciblent les utilisateurs n'ayant aucune connaissance en solfège en musique comme SongSmith (Simon *et al.* (2008)). À un autre extrême, réservé à un public de connaisseurs, nous pouvons trouver des langages de programmation textuels (AthenaCL de Ariza (2005)) ou visuels (Openmusic et Patchwork (Assayag *et al.* (1999))) qui permettent une grande flexibilité dans le processus de création. Mais cette puissance a un prix : la courbe d'apprentissage est raide, le processus de création est souvent rendu long et complexe. Il arrive même que certains compositeurs développent leurs propres applications afin de répondre à des besoins très spécifiques.

Nous nous sommes demandé dans un premier temps s'il pouvait exister une solution pour pouvoir générer rapidement des mélodies, puis de pouvoir écouter rapidement ces mélodies générées, les visualiser, et pouvoir faire le choix de les conserver, de les supprimer, de les copier, de les modifier, voire même de les réutiliser dans les générations suivantes en conservant certaines informations comme certaines notes ou bien le rythme d'une séquence. Nous voulions permettre à l'utilisateur de visualiser un historique grâce à des liens de parenté entre les séquences musicales : par exemple, nous voulions montrer que telle séquence avait été générée à

partir de telle séquence ou avait été copiée de telle autre. Par la suite nous nous sommes demandés comment représenter les paramètres des algorithmes de génération. Comment permettre à l'utilisateur de configurer un nombre important de paramètres (peut-être 10) et de nature varié (chaine de caractère, booléen, entier, flottant), tout en donnant à l'utilisateur la possibilité de visualiser quels paramètres ont été utilisés pour générer telle séquence ? Cela implique de conserver les paramètres en mémoire, d'imaginer un système de visualisation et d'éditions de ces paramètres, de créer un système de visualisation des liens entre les paramètres et les séquences générés, de donner à l'utilisateur la possibilité de réutiliser les paramètres ayant servi pour générer une séquence pour en générer de nouvelles. Nous voulions aussi donner à l'utilisateur la possibilité de mettre les séquences générées bout à bout pour obtenir des séquences plus longues, mais aussi de réutiliser les séquences générées dans d'autres logiciels d'aide à la composition.

De plus, nous pensions qu'un tel système devrait être accessible au plus grand nombre. Nous avons classé les utilisateurs en trois groupes de niveaux : les *débutants* qui ont une connaissance très limitée de la théorie musicale ; les *intermédiaires* qui ont une expérience significative de la pratique musicale et une maitrise des notions de base du solfège ; les *confirmés* qui ont une connaissance précisent du solfège et une bonne expérience en tant que musicien ou compositeur. Nous avions émis l'hypothèse que les utilisateurs *confirmés* ne sont pas les premiers concernés par l'utilisation potentielle d'un système de composition automatique. Pour répondre aux besoins potentiels des utilisateurs débutants, il fallait que l'application soit la plus intuitive dans son utilisation, notamment avec le système de visualisation des séquences et des paramètres sous forme d'objets, et avec le système d'édition des objets et le système de transition entrent les différentes vues (vues d'édition et vue globale des objets). Nous pensions que les utilisateurs débutants et intermédiaires seraient les plus intéressés par un tel système de génération de mélodies. Et nous pensions que plus le système permettrait des réglages fins du générateur, et donc gérerait un nombre conséquent de paramètres, plus le système serait susceptible de satisfaire des compositeurs ayant de l'expérience. Il était donc important de laisser la possibilité à l'utilisateur de contrôler un maximum de paramètres, dans la mesure du possible. Aussi, les utilisateurs *intermédiaires* comblent tout l'espace de possible entre les

niveaux de connaissance des utilisateurs débutants et confirmés. Nous pensions qu'ayant une connaissance, même faible, du solfège et de l'écriture, ils apprécieraient avoir une sensation de contrôle des résultats sans que les paramètres et les outils d'édition soient trop complexes à appréhender.

Nous avons alors décidé de développer un prototype avec une interface utilisateur permettant d'explorer l'utilisation de générateurs de mélodies à l'aide d'une approche novatrice permettant de conserver et de visualiser l'historique des paramètres de générations utilisés, d'organiser et éditer les résultats des générations, de réutiliser les résultats des générations dans de nouvelles générations. Le système permettra de naviguer rapidement entre la vue globale de l'ensemble des séquences générées, la vue d'édition d'une séquence et la vue permettant le choix des paramètres de génération. Le système donnera des résultats et offrira une utilisation pouvant satisfaire des utilisateurs débutants et intermédiaires. Le prototype s'appellera GenSession.

Ce projet de recherche s'est déroulé de la manière suivante : après avoir effectué une recherche documentaire sur les travaux antérieurs, nous avons testé nombre de prototypes de recherche et de logiciels commerciaux. De cette analyse des forces et faiblesses des solutions existantes, nous avons défini notre problématique en mettant en lumière la nécessité de développer et d'évaluer un prototype logiciel. Nous avons ensuite entrepris le développement d'un algorithme de génération en parallèle du développement d'une interface utilisateur. Après 1 an de travail, une fois une ébauche du prototype développé, nous somme entré en contact avec des membres du laboratoire CIRMMT (Centre for Interdisciplinary Research in Music Media and Technology) de McGill. Nous avons eu plusieurs rencontres avec le professeur Marcelo M. Wanderley et le doctorant Marlon Schumacher. Nous avons pris en compte leurs suggestions dans la suite du développement. Ils nous ont aussi, par la suite, guidés dans le choix de la conférence et ont contribué à l'écriture de notre article (Cabrol *et al.* (2013)). Une fois le développement du prototype terminé, nous avons effectué des évaluations qualitatives auprès de sept utilisateurs, dont cinq membres du CIRMTT.

Lors de la conception de GenSession et afin de visualiser les séquences musicales générées, nous avons choisi de représenter ces séquences sous forme de rectangles disposés sur un es-

pace de travail en 2 dimensions. Nous avons décider d'appeler ces objets représentés sous forme de rectangles des $clips$ en référence au clip du logiciel Ableton Live (Ableton (2013)). Des liens seraient dessinés entre ces objets et représenteraient les liens de parenté entre les séquences musicales (par exemple, un lien entre deux de ces rectangles peut représenter la copie). Ces liens devraient permettre de visualiser l'historique des générations, des créations et des copies de clips. Nous voulions donner à l'utilisateur la possibilité d'éditer les séquences générées, et pour pouvoir éditer les séquences nous avons pensé qu'il fallait une vue d'édition. Nous voulions un système qui puisse permettre de passer rapidement d'une vue détaillée de l'objet, permettant de visualiser son contenu et de le modifier, à une vue globale qui permet de visualiser tous les objets et les liens qu'il y a entre eux. Nous avons alors pensé à utiliser le principe des ZUI tels que dans le travail de Bederson et Hollan (1994). Un système de zoom automatique permettrait de naviguer rapidement entre la vue globale et la vue d'édition d'un clip, avec une animation de zoom. Nous avons appelé *scène*, la vue globale.

Par la suite, la question de la visualisation et de l'édition des paramètres de l'algorithme de génération s'est posée. Comment faire pour visualiser les paramètres de génération ayant été utilisés pour générer tel ou tel $clips$, pouvoir les dupliquer, les éditer et pouvoir les réutiliser pour la génération de nouvelles séquences musicales ? Nous avons imaginé représenter les paramètres de génération par des objets positionnés sur la *scène*, semblable aux objets $clip$. Ces objets représentant les paramètres de génération pourraient être créés, copiés, supprimés et pourraient servir à plusieurs exécutions du générateur. Des liens entre les objets *paramètres de génération* et les objets $clips$ permettraient de visualiser l'historique des générations : on visualiserait alors le lien entre les paramètres et les séquences qu'ils auront permis de générer. Le système de zoom automatique nous permettra de naviguer entre la vue globale de la *scène* et la vue d'édition des objets *paramètres de génération*. La vue d'édition de ces objets permettra à l'utilisateur de configurer des *paramètres globaux* qui s'appliqueront à l'ensemble de la séquence générée et lui permettra aussi de dessiner des courbes représentant des *paramètres dynamiques* qui évoluent dans le temps.

Afin de soutenir le développement de l'interface, nous avons fait le choix de développer un algorithme de génération basé sur les règles, inspiré du prototype de Povel (2010). Nous avons ajouté des caractéristiques telles que la possibilité d'utiliser des *paramètres dynamiques*, qui varie dans le temps. Ainsi chaque unité de temps de la séquence générée, le paramètre peut prendre une valeur différente.

GenSession permet d'automatiser une partie du processus de création de mélodies tout en gardant toujours l'humain au centre du processus en lui offrant le choix des contraintes (paramètres), des algorithmes de génération, et l'évaluation des résultats de chaque génération. De plus, notre système permet à l'utilisateur d'éditer les mélodies générées, mais aussi de créer des mélodies manuellement. Il se situe donc entre les logiciels d'aide à la composition et les logiciels de composition automatique.

Nos principales contributions sont au nombre de trois. La première contribution est l'utilisation d'une interface utilisateur zoomable pour naviguer au travers d'une représentation graphique du réseau relationnel liant les $clips$ et les paramètres de génération. La repésentation et la sauvegarde de paramètres de génération sous forme d'objets est une approche nouvelle. Les liens entre les objets permettent de visualiser l'historique des générations alors que le zoom automatique permet de passer d'une vue globale à une vue d'édition des séquences musicales ou des paramètres de génération. La deuxième contribution est la possibilité de dessiner des courbes pour définir des paramètres de génération dynamiques et la troisième est une évaluation préliminaire avec des sujets étudiants et chercheurs qui confirme partiellement la valeur des caractéristiques de notre prototype, dont son accessibilité pour les utilisateurs débutants et intermédiaires.

Dans ce mémoire sera d'abord présenté un état de l'art, de la recherche, mais aussi des solutions commerciales, dans le domaine des applications d'aide à la composition et de composition automatique. Après avoir fait le tour des solutions existantes, nous présenterons notre problématique. Puis nous présenterons notre solution avec une description de notre prototype logiciel GenSession, de son interface utilisateur et de ses algorithmes de génération. Ensuite, nous par-

lerons de la façon dont nous avons évalué la solution proposée. Pour finir, nous présenterons les résultats de l'évaluation et discuterons ces résultats avant de conclure.

Le code Java de notre prototype est en ligne sur la page web de présentation du projet : http://hifiv.ca/~francoiscabrol/GenSession/, Cabrol (2013). On y trouve également une vidéo de démonstration de l'interface utilisateur et une version compilée téléchargeable du prototype.

Ce mémoire a donné lieu à la publication d'un article, Cabrol *et al.* (2013), à la conférence internationale CMMR 2013 (10th International Symposium on Computer Music Multidisciplinary Research).

CHAPITRE 1

LA REVUE DE LITTÉRATURE

Cette revue de littérature est un état de l'art dans le domaine des algorithmes de génération de mélodie et des interfaces homme-machine permettant leur utilisation. Tout d'abord sont présentées les grandes catégories d'algorithmes qui permettent de générer de la musique. Nous nous intéressons plus particulièrement ici à ceux qui écrivent des séquences musicales au format MIDI pouvant être ensuite traduites pour être lues par des musiciens humains aussi bien que par des séquenceurs numériques ou des instruments virtuels.

1.1 Les logiciels d'aide à la composition et de composition automatique

La différence entre la composition aidée par ordinateur et la composition automatique est exprimée dans Anders *et al.* (2003). La première a pour vocation d'assister le compositeur humain dans le processus de création alors que la seconde automatise le processus en excluant l'humain de l'exécution de ses différentes tâches.

Des outils permettent donc d'accompagner les musiciens dans leur travail et parfois de les remplacer dans l'exécution de certaines tâches. La calculatrice a changé la façon dont les mathématiciens font des calculs numériques leur permettant à la fois d'aller plus vite dans l'exécution de certaines tâches, d'en exécuter d'autres auparavant impossibles ou très pénibles avec facilité, et parfois cet outil les a même remplacés entièrement dans l'exécution de certaines activités comme le traçage d'une fonction (pour ensuite la visualiser) ou bien encore pour faire le calcul de π. L'idée est répandue selon laquelle faire de la musique c'est faire des mathématiques sans le savoir et que comprendre l'un peut permettre d'acquérir une logique apportant une meilleure compréhension de l'autre. La musique peut être vue comme un ensemble de phénomènes physiques pouvant être représentés par des modèles mathématiques. Un lien existe donc entre mathématiques et musique. L'admettre est important pour comprendre comment les outils numériques du musicien peuvent l'assister, voire le remplacer, dans l'exécution de certaines tâches d'un processus de création tout comme la calculatrice peut assister les mathéma-

ticiens au cours du processus de résolution de problèmes. Sans doute ne pourrons-nous jamais voir d'innovation dans les mathématiques sans mathématiciens, bien que ce soit sujet à débat (Ruelle (2008)), tout comme nous ne verrons sans doute pas l'émergence de nouvelles musiques sans compositeurs. Mais nous avons vu et verrons sans nul doute l'arrivée de nouveaux outils qui rendront possible l'automatisation de nombreuses activités du processus de composition au sens large. Gardons en tête que chaque compositeur a son(ses) propre(s) processus de création et c'est aussi ça qui fait l'originalité de sa production. En attendant, intéressons-nous aux technologies qui existent déjà.

1.1.1 Le MIDI

Quand on parle d'un processus automatique qui génère de la matière musicale, celle-ci peut prendre différentes formes. Sans entrer dans les détails, il nous semble important de faire le point sur le format le plus utilisé pour transmettre des informations musicales au format numérique qui est aussi le format que nous avons utilisé pour lire et exporter les séquences musicales de notre prototype.

MIDI signifie Musical Instruments Digital Interface. La norme MIDI est à la fois un protocole de communication permettant l'échange de donnés musicales en temps réel entre la plupart des instruments virtuels, électroniques, et séquenceurs sous forme de messages MIDI, ainsi qu'un format de fichier pouvant contenir ces messages MIDI accompagnés d'informations temporelles. Ces fichiers peuvent être importés ou exportés dans la majorité des logiciels de musique et peuvent ensuite permettre de visualiser ces informations sous forme d'une partition par exemple, aussi bien que d'être incorporés sur la piste d'un projet dans un séquenceur logiciel.

Il existe une autre norme, l'OSC, ayant la même finalité que le MIDI et ayant pour vocation de la remplacer. En effet, l'OSC est bien plus récente (la norme MIDI date de 1983) et est plus riche que son prédécesseur. Elle est de ce fait souvent favorisée par la communauté scientifique, depuis quelques années, dans l'expérimentation de nouveaux prototypes logiciels. Cependant, les solutions commerciales sont pour l'essentiel encore compatible nativement exclusivement

avec la norme MIDI. Si nous avons choisi le MIDI c'est pour sa compatibilité avec la majorité des systèmes existants et pour la gestion facilitée de celle-ci par l'API java.sound de Java. De plus, elle répond entièrement à nos besoins actuels. Si ceux-ci venaient à évoluer, il serait possible d'implémenter la gestion de la norme OSC, en plus de celle du MIDI.

La principale différence entre le MIDI et l'OSC est que le MIDI est originellement prévu pour être communiqué entre les systèmes au travers d'un câble MIDI physique alors que l'OSC est prévu pour être transmis au travers d'un réseau informatique. Pour transmettre du MIDI entre deux applications logicielles, par exemple, il faut créer un port MIDI « virtuel » à l'aide d'un petit outil logiciel, ce qui est (un peu) contraignant puisque non géré nativement.

1.1.2 La MAO et les séquenceurs

La MAO c'est la Musique Assistée par Ordinateur. Tout usage de l'informatique pour faire de la musique, c'est de la MAO. Les outils de MAO peuvent ainsi prendre des formes extrêmement diverses pour soutenir les divers processus de composition, de production, de sonorisation, mais aussi bien être des outils d'apprentissages de la musique, de technique de production, etc.

À l'origine, les séquenceurs permettaient de manipuler des séquences MIDI, de les arranger, de rediriger le signal vers d'autres systèmes prenant la norme MIDI (voir section 1.1.1) en entrée telle que des synthétiseurs, expandeurs ou échantillonneurs et d'acquérir des messages provenant d'autres systèmes. Aujourd'hui, la plupart des séquenceurs permettent aussi de gérer de l'audio, d'intégrer des applications prenant en entrée du midi telle que des synthétiseurs virtuels, de rediriger les signaux audio et les messages MIDI vers tout type de système matériel ou logiciel compatible et de récupérer le MIDI et l'audio de la même façon en entrée. On les appelle des DAW (vient de l'anglais Digital Audio Workstation qui signifie station audionumérique). Bien que l'acronyme DAW est censé désigner l'ensemble des outils d'une configuration pour la production de musique (ou audiovisuelle), ces séquenceurs, dont nous parlions, proposent un ensemble d'outils tellement complet qu'on les nomme ainsi. Le plus souvent, les DAW logiciels s'accompagnent de cartes son pour s'interfacer avec des systèmes

matériels (un microphone en entrée pour prendre un instrument et des enceintes amplifiées en sortie par exemple) et de surfaces de contrôle (principalement MIDI).

Pour plus d'informations à propos des outils de MAO actuellement utilisés pour la composition de musique, il peut être intéressant de se référer à la thèse Han (2011) qui a pour vocation d'étudier des méthodes de composition utilisant les outils les plus vendus. Bien que les technologies évoluent rapidement, ce travail est suffisamment récent pour être actuel. Jinseung Han s'attarde notamment à détailler les fonctionnalités de Pro Tools (Digidesign (1991-2013)) et Cubase (Steinberg (1990-2012)) qui sont les deux séquenceurs les plus répandus du marché professionnel.

Si l'on cherche des informations sur les outils matériels et logiciels utilisés pour la composition, mais aussi pour tout ce qui touche de près ou de loin à la musique et au son, l'une des plus importantes bases de données francophones est, à notre connaissance, celle du fanzine web Audiofanzine (Raynaud (2000-2013)). Cette source d'information est très riche, mais attention aux sources qui alimentent le contenu, celui-ci est en partie contributif.

En 1999, le logiciel Ableton Live (Ableton (2013)) est un nouveau séquenceur proposant une approche nouvelle pour la manipulation de séquence audio et MIDI sous forme clip. Un clip c'est une séquence d'une ou plusieurs mesures, d'un ou plusieurs temps. Cette séquence peut être jouée en boucle ou bien une seule fois. La lecture d'un clip peut être lancée rapidement et plusieurs clips peuvent être lus simultanément. L'ordre de lecture des clips peut être automatisé. Les clips sont placés dans une grille dans une vue appelée $session$. Ce que vous voyez sur la figure 1.1 est un exemple de configuration en vue session. Les clips sont représentés par des rectangles de couleurs avec un petit bouton « lecture » sur la gauche. Lorsqu'un clip est sélectionné, son contenu apparait en bas de la fenêtre. Une vue $arrangement$ permet d'enregistrer l'exécution d'une session en live pour ensuite la travailler dans un séquenceur plus traditionnel. Cette approche d'Ableton Live est très appréciée des musiciens de musique électronique tant pour avoir offert de nouvelles possibilités lors des performances scéniques que pour donner un aspect vivant à leur processus de composition et de production en studio. Bien que plus adapté à la production de musique électronique et surtout répétitive que pour la prise d'instruments

Figure 1.1 L'interface d'Ableton Live
Imprime-écran de (Ableton, 2013)

acoustiques, il permet de s'aventurer dans n'importe quel genre autant pour la composition que pour la production.

1.1.3 SongSmith

SongSmith est un logiciel de composition automatique qui permet de générer un accompagnement pour une mélodie chantée dans un microphone par l'utilisateur. Initialement, le prototype s'appelait MySong et est issu du travail (Simon *et al.* (2008)). La Figure 1.2 montre l'interface de MySong. La génération de l'accompagnement est rendue possible par un système d'harmonisation automatique utilisant le principe d'un modèle de Markov caché avec pour mémoire musicale une base de données de musique dont le contenu provient en partie de la base de données en ligne Wikifonia (2013).

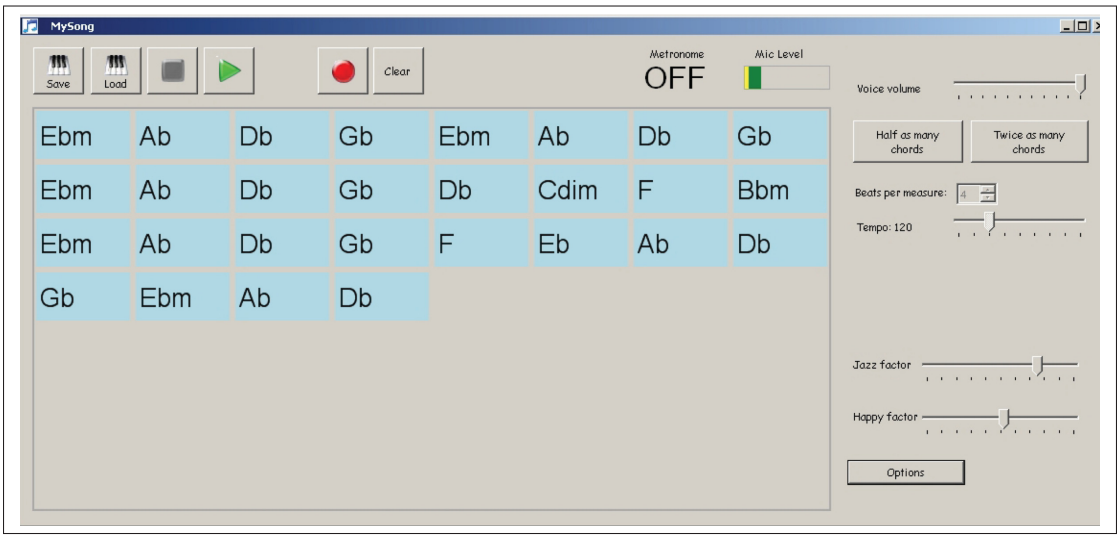

Figure 1.2 L'interface de MySong
Tiré de (Simon *et al.*, 2008)

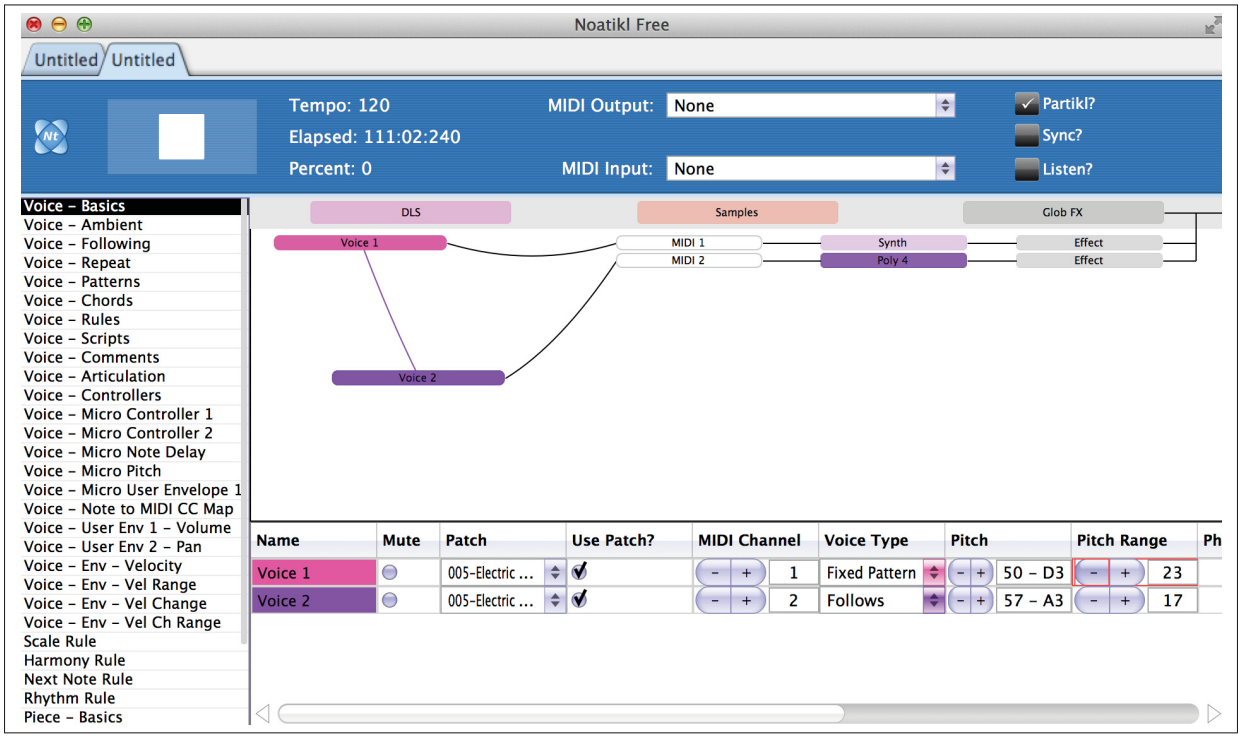

Figure 1.3 La vue principale du logiciel Noaktil
Imprime-écran de (Intermorphic, 2013)

1.1.4 Noaktil

Noaktil 2 de Intermorphic (2013) est une application qui se veut un «laboratoire de musique générative». L'approche est intéressante puisqu'Intermorphic nous propose de générer aléatoirement, en temps réel, à la fois des informations musicales sous la forme d'un flux de messages MIDI ainsi que de rédiger ce flux vers un synthétiseur intégré dans l'application. Cette chaine MIDI-audio est représentée sous la forme d'un ensemble d'objets qui offrent de l'audio en sortie. Cet ensemble est constitué d'un objet Synth qui représente le synthétiseur et l'objet Effect permettant de moduler le signal. Le signal est ensuite redirigé vers la sortie de l'application. En entrée du système de génération, on a donc les paramètres choisis par l'utilisateur et en sortie un flux audio généré aléatoirement. L'interface est assez complexe à utiliser et pour modifier les paramètres de génération il faut fouiller pas mal dans les menus. On ne peut choisir de générer sur un accord ou une gamme, mais on ne peut pas définir de progression harmonique, sur plusieurs mesures par exemple.

La Figure 1.3 nous montre la vue principale de Noaktil sur laquelle on peut voir que les différents éléments de la « chaine de génération » sont représentés par des rectangles de couleurs liés entre eux par des câbles noirs. On peut y voir deux objets $voice$ 1 et 2 au centre. Dans le panneau du bas, pour chacune des $voices$, certains paramètres de génération du flux de messages midi sont éditables.

1.1.5 Hyperscore

Voilà une interface fort intéressante, car les concepteurs ont mis l'accent sur l'accessibilité, mettant leur effort au service des personnes n'ayant pas nécessairement de connaissances en musique ou d'expérience en composition. Ces expérimentations ouvrent, semble-t-il, de nouvelles voies d'expressions, qui peuvent stimuler la créativité.

Même s'il s'agit aujourd'hui d'une application commerciale, Hyperscore (Morwaread *et al.* (2007)) était à l'origine le prototype d'un projet de recherche promettant d'offrir une expérience utilisateur originale en permettant à des non-initiés de composer une pièce de musique.

L'interface propose de suivre un processus qui consiste dans un premier temps à positionner sur un espace de travail des objets représentant de courtes mélodies pré-écrites, des boucles rythmiques pré-écrites, assigner une couleur à chacun de ces éléments, puis de les positionner sur une ligne de temps en dessinant des courbes dans les couleurs correspondantes sur un objet unique prévu à cet effet. C'est autour du concept de cet objet central qui représente le lien temporel et harmonique entre les passages musicaux superposables que repose l'originalité de l'interaction.

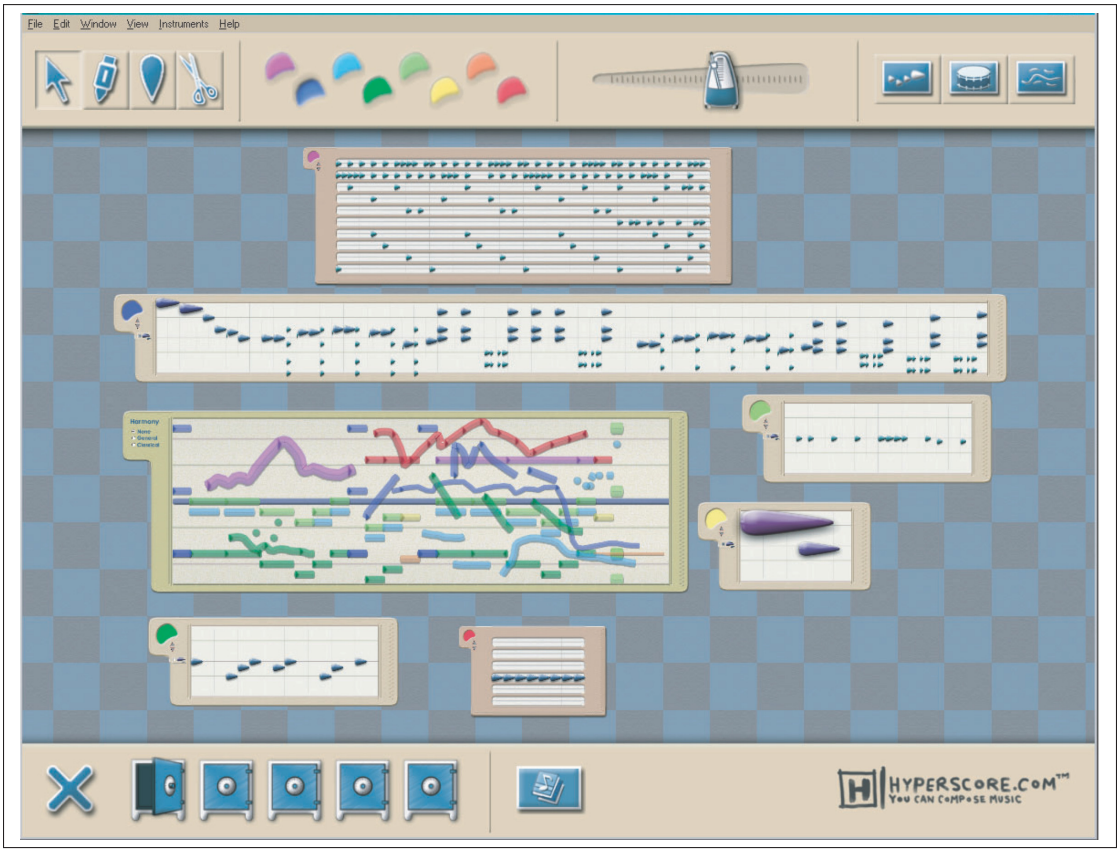

Figure 1.4 L'interface de Hyperscore
Tiré de (Morwaread *et al.*, 2007)

La Figure 1.4 nous montre la vue principale d'Hyperscore. Comme Morwaread *et al.* (2007) le disent « Users can position these musical objects anywhere on the canvas and can view the workspace at any level of zoom for ease of editing. ». Certains détails dans les objets peuvent apparaitre dépendamment du niveau de zoom ce qui en fait une interface zoomable partageant

certains concepts propres au ZUI (voir section 3.2.1). Il n'y a pas cependant de fonctionnalité permettant de zoomer automatiquement sur les objets et l'espace de travail où prennent place les objets est un espace limité qui ne peut être étendu. Les outils d'édition ne sont pas non plus très évolués, l'accent étant mis sur la fonction de placement et de superposition des différents passages rythmiques et mélodiques dans le temps. Hyperscore permet donc, à des utilisateurs de tous niveaux de créer des morceaux de musique à partir d'une bibliothèque de passages mélodiques et rythmiques déjà écrits. La marge de manœuvre de l'utilisateur est restreinte par la diversité de la base de données des passages musicaux déjà écrits et l'interface a un aspect enfantin qui la destine principalement à un usage éducatif.

1.1.6 Pizzicato

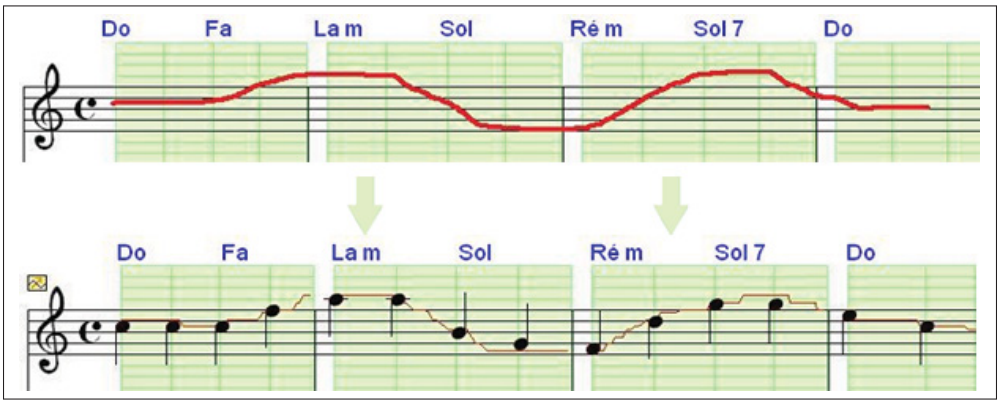

Figure 1.5 L'interface de Pizzicato
Tiré de (Vandenneucker, 2011)

Depuis 1992, la société Arpège Musique développe Pizzicato (Vandenneucker (2011)), un logiciel commercial d'aide à la composition un peu particulier, car il se destine à la fois à l'éducation pour les étudiants en conservatoire et à la composition, plutôt adapté aux musiciens et compositeurs classiques. Sur le site, la société dit avoir commencé la conception du logiciel après une étude de marché auprès des cibles potentielles permettant de mieux comprendre leurs objectifs. Nous sommes en 1993.

> La première constatation était qu'il n'y avait pas de programme qui combinait astucieusement un outil professionnel avec une approche didactique et accessible

> au débutant en musique comme en informatique. [...] La seconde constatation était que les grands logiciels musicaux du moment ne semblaient pas (ou peu) toucher à l'aide à la composition, c'est-à-dire à l'idée que l'ordinateur pourrait en plus de simplement écrire et imprimer ce qu'on lui demande, nous proposer des solutions d'aide à la composition, au choix des accords, à la création de thèmes musicaux [...] Notre objectif était donc fixé [...] (Arpège Musique, 2013)

Aujourd'hui, Vandenneucker (2011) poursuit ses objectifs et nous a livré avec la dernière version de son logiciel quelques fonctionnalités qui démarquent clairement le produit de la concurrence. La Figure 1.5 illustre l'une d'entre elles : il s'agit de la possibilité de dessiner des courbes qui représentent ce qu'ils appellent des « vecteurs musicaux ». Ces vecteurs sont ensuite traduits pour définir la hauteur des notes ou le rythme. Par exemple, pour la hauteur des notes, le vecteur musical représente ce qui est appelé aussi parfois la courbe mélodique. Si on relie chaque note, de la partition écrite dans la notation standard, on obtient cette courbe. L'idée est de faire l'inverse, de dessiner la courbe pour obtenir les notes. Les hauteurs des notes alors écrites sont les hauteurs des notes autorisées par l'harmonie, les gammes et les progressions d'accords précédemment définies, les plus proches des points de la courbe à l'unité de temps où elles sont positionnées.

1.1.7 L'intégration des générateurs de mélodies dans les DAW

Les logiciels d'aide à la composition sont nombreux et prennent différentes formes pour répondre à de nombreux besoins et s'adapter à de nombreuses approches. Pourtant nous avons pu constater que les applications commerciales d'aide à la composition utilisant des algorithmes de génération de séquence musicale étaient rares, relevant plus de l'expérimentation pour la plupart que du produit grand public, du moins jusqu'à récemment. Il y a trois ou quatre ans, il aurait été difficile de citer plus de trois applications commerciales de musique générative, générant des mélodies, offrant une solution complète pour les compositeurs. Aujourd'hui, il en existe et des outils de composition automatique trouvent leur place dans les logiciels les plus utilisés (par exemple le $DAW\ logiciel$ FLStudio et son «randomizer», Image-Line (2013)). Celui-ci permet de générer des mélodies très simples sur une gamme ou un accord, en plusieurs étapes.

1.2 Les algorithmes de génération de musique

> Systems based on only one method do not seem to be very effective. We could conclude that it will become more and more common to 'blend' different methods and take advantage of the strengths of each one. (Papadopoulos et Wiggins, 1999)

Il y a quatre grandes catégories d'algorithmes qui écrivent de la musique : les algorithmes génétiques, les algorithmes stochastiques basés sur les règles de composition, les algorithmes basés sur les chaines de Markov, les algorithmes déterministes (basé sur des langages de programmation de logique sous contraintes). Dans cette section, nous allons principalement discuter des algorithmes génétiques et des algorithmes basés sur les règles.

1.2.1 Les algorithmes génétiques

Le principe des algorithmes génétiques pour la génération de musique vient de la bio-informatique. En bio-informatique, on l'utilise pour simuler l'évolution du matériel génétique d'une population à travers les générations d'individus. Chaque reproduction simulée entraine un processus itératif qui simule les mutations génétiques que subit l'ADN des individus parents pour devenir celui des enfants. La somme de ces itérations représente l'évolution. À chaque itération, certains individus sont choisis pour survivre au cycle, tandis que d'autres sont éliminés, par un algorithme appelé fonction d'évaluation (en anglais « fitness function »). Ceci simule l'évolution naturelle.

En musique générative, des chercheurs ont démontré qu'il est possible d'utiliser ce principe pour faire évoluer des « idées » musicales, qui peut se situer a plusieurs niveaux du processus de composition (création de mélodies, arrangement, choix de progression d'accords, choix/-création du timbre ...) ou sur son ensemble. Cette matière peut être un bout de partition qui pourrait servir à composer ou encore être un passage musical audio riche constituant une fin. L'évaluation peut-être exécutée par un algorithme. Mais des résultats obtenus ont démontré que la musique générée par les algorithmes génétiques dont l'évaluation est faite par des algorithmes offre des résultats moins satisfaisants pour l'utilisateur (Keup (2011)) que lorsqu'elle est faite par des humains, à moins que l'on souhaite générer une musique dans le style d'un

compositeur ou d'un improvisateur en particulier. La plupart des recherches se concentrent sur des algorithmes génétiques avec une évaluation humaine, pour ce qui est de la composition. À chaque itération, un humain fait le choix de garder ou non chaque résultat de la génération. Des essais ont été effectués sur des groupes finis d'individus comme Unehara et Onisawa (2003). Un problème soulevé est qu'il faut de nombreuses itérations et donc de nombreuses évaluations pour obtenir des résultats convaincants. Mais dernièrement, des résultats très satisfaisants ont été obtenus avec du « crowdsourcing » avec le prototype Darwin Tunes (MacCallum *et al.* (2012)) : les résultats de chaque génération sont diffusés sur une plateforme web. N'importe quelle personne qui se connecte à l'application peut devenir évaluateur pour un instant. Ainsi les itérations peuvent être très nombreuses en peu de temps (selon la fréquentation de l'application web) et les cycles peuvent s'enchainer jour et nuit, sur une période indéterminée. Notons que dans les travaux d'Unehara et Onisawa, les mutations s'appliquent sur des passages de musique en MIDI tandis que pour le projet Darwin Tunes celles-ci travaillent des sons au format audio. En fonction des projets, les mutations génétiques peuvent ainsi être de nature très différente. On retrouve cependant assez souvent des algorithmes typiques tels que ceux proches du concept de « crossover » où l'on mélange deux passages musicaux (par exemple deux mélodies monophoniques de deux mesures) pour en obtenir une nouvelle.

Dans Biles (1994), GenJam est un algorithme qui improvise sur une progression d'accords. Celui-ci doit être entrainé préalablement, c'est-à-dire qu'il apprend par interaction avec un humain. Le programme génère des phrases musicales, les fait écouter à l'utilisateur et prend en compte son feedback pour décider si, oui ou non, celles-ci sont conservées dans sa base de données. Il peut ensuite puiser dans cette base de données lorsqu'il improvise. Biles a procédé à de nombreuses performances durant lesquelles il joue des standards de jazz accompagner de (ou accompagnant) son système.

D'un autre côté, de nombreux travaux très intéressants utilisent des fonctions algorithmiques pour évaluer les résultats de chaque itération, l'évaluation par un humain peut se produire après l'exécution d'un certain nombre d'itérations. Les chaines de Markov sont souvent utilisées dans les fonctions d'évaluation comme nous le montrent Lo et Lucas (2006) .

1.2.2 Les algorithmes non déterministes basés sur les règles

Tout algorithme non déterministe basé sur les règles peut être dit stochastique. Stochastique est un synonyme d'aléatoire, mais, à notre sens, il met tacitement l'accent sur l'existence de contraintes qui influencent le résultat. On parle souvent d'algorithmes stochastiques pour parler d'algorithmes dont le résultat est influencé par des valeurs choisies de manière pseudo-aléatoire. Nous nous intéressons ici à des algorithmes stochastiques dont les valeurs de retour sont déterminées par des paramètres, par des contraintes et par des nombres générés avec un générateur de nombre aléatoire (ou pseudo-aléatoire).

Gwee (2002) est un travail très dense de théorisation de l'utilisation de ce type d'algorithme pour générer des mélodies à contrepoints d'un cantus firmus. Le cantus firmus, c'est une phrase qui revient plusieurs fois dans une pièce de musique, c'est celle qui marque le plus et que l'on retient. Ces algorithmes peuvent utiliser de nombreux concepts d'apprentissage machine et de probabilités tels que les réseaux neuronaux, les marches aléatoires, les chaines de Markov, ou bien encore les algorithmes génétiques.

Le prototype « Melody Generator » de Povel (2010) fonctionne sur ce principe stochastique. Il s'agit d'un générateur de mélodie tonale. Une interface permet de choisir les paramètres d'entrées de l'algorithme et de visualiser les mélodies ainsi que de les générer les unes à la suite des autres pour construire une suite mélodique sur un nombre n de mesures. L'interface n'est pas le sujet central de ce projet, mais permet cependant une utilisation du générateur simplifié et grandement accessible, bien que nécessitant un temps d'apprentissage et des notions en musique. Les paramètres de génération sont nombreux et offrent une liberté ainsi qu'une précision permettant d'obtenir ce que l'on veut sans pour autant déterminer le résultat. Des degrés de liberté donnés à l'algorithme peuvent être ainsi définis dans certains cas comme, par exemple, pour la courbe mélodique définissant le contour mélodique. Cette courbe peut être choisie sinusoïdale, ascendante, descendante, ou encore totalement aléatoire. C'est une sorte de degré de déterminisme qui peut alors être défini pour certains paramètres. L'interface de MGII est montrée par la Figure 1.6.

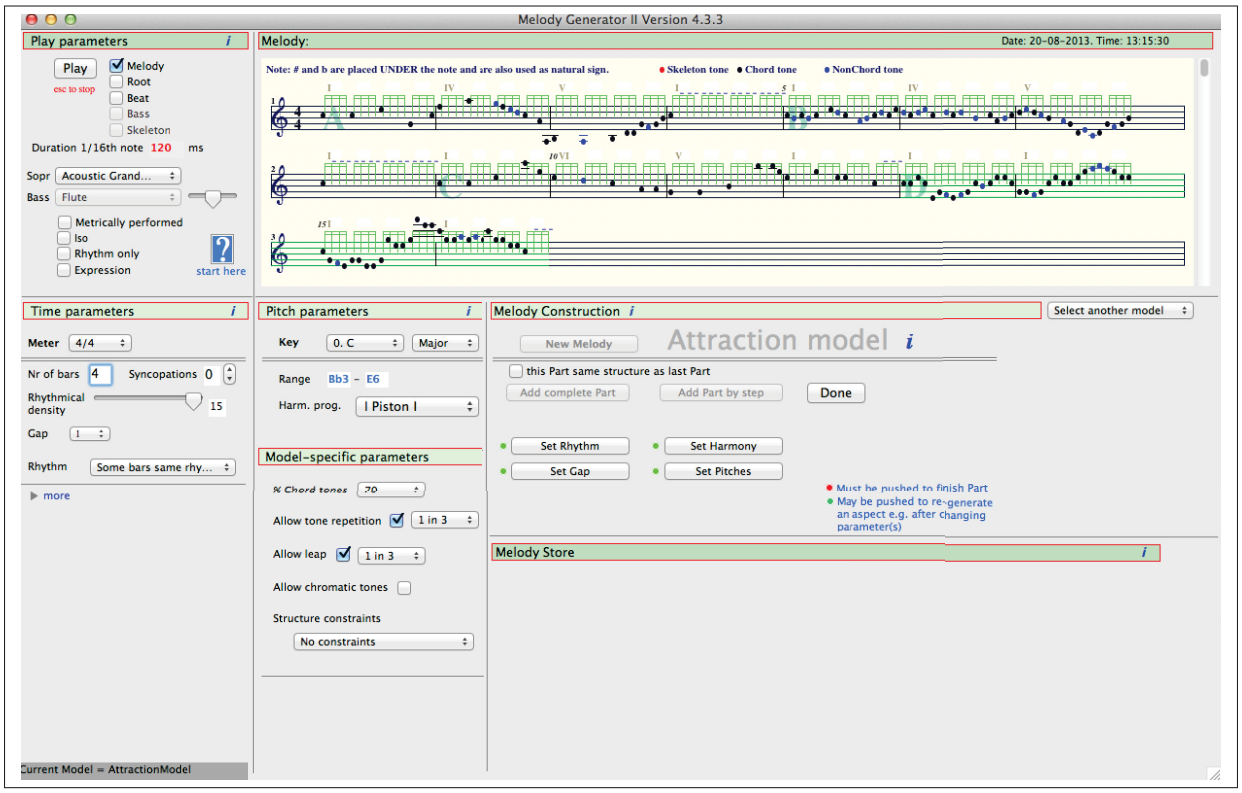

Figure 1.6 L'interface du prototype Melody Generator II
Tiré de (Sweers *et al.*, 2012)

De nouvelles règles sont implémentées et testées, dans le mémoire de baccalauréat de Sweers *et al.* (2012), et « Melody Generator » passe à une version II. En conclusion de ce mémoire, la phrase suivante est éloquente : Sweers *et al.* (2012, p. 8) « The MG II shows potential, with new rules it might compose melodies just as well as humans. »

Le système APOPCALEAPS, de Sneyers et De Schreye (2010), est un système de génération basé sur le langage de programmation pour apprentissage de logique probabiliste nommé CHRISM (CHance Rules induce Statistical Models) de Sneyers *et al.* (2010). Lui-même est une combinaison du langage de gestion des contraintes CHR (Constraint Handling Rules) et de l'extension probabiliste de Prolog (principal langage de programmation logique) nommé PRISM (PRogramming In Statistical Modeling). Un programme CHRISM est une séquence de règles probabilistes. Étant une bonne approche pour développer un algorithme de génération basé sur les règles, CHRISM est selon les auteurs Sneyers et De Schreye (2010, p. 7) « un

excellent langage de programmation pour la composition automatique [...] ». La Figure 1.7 montre l'interface d'APOPCALEAPS.

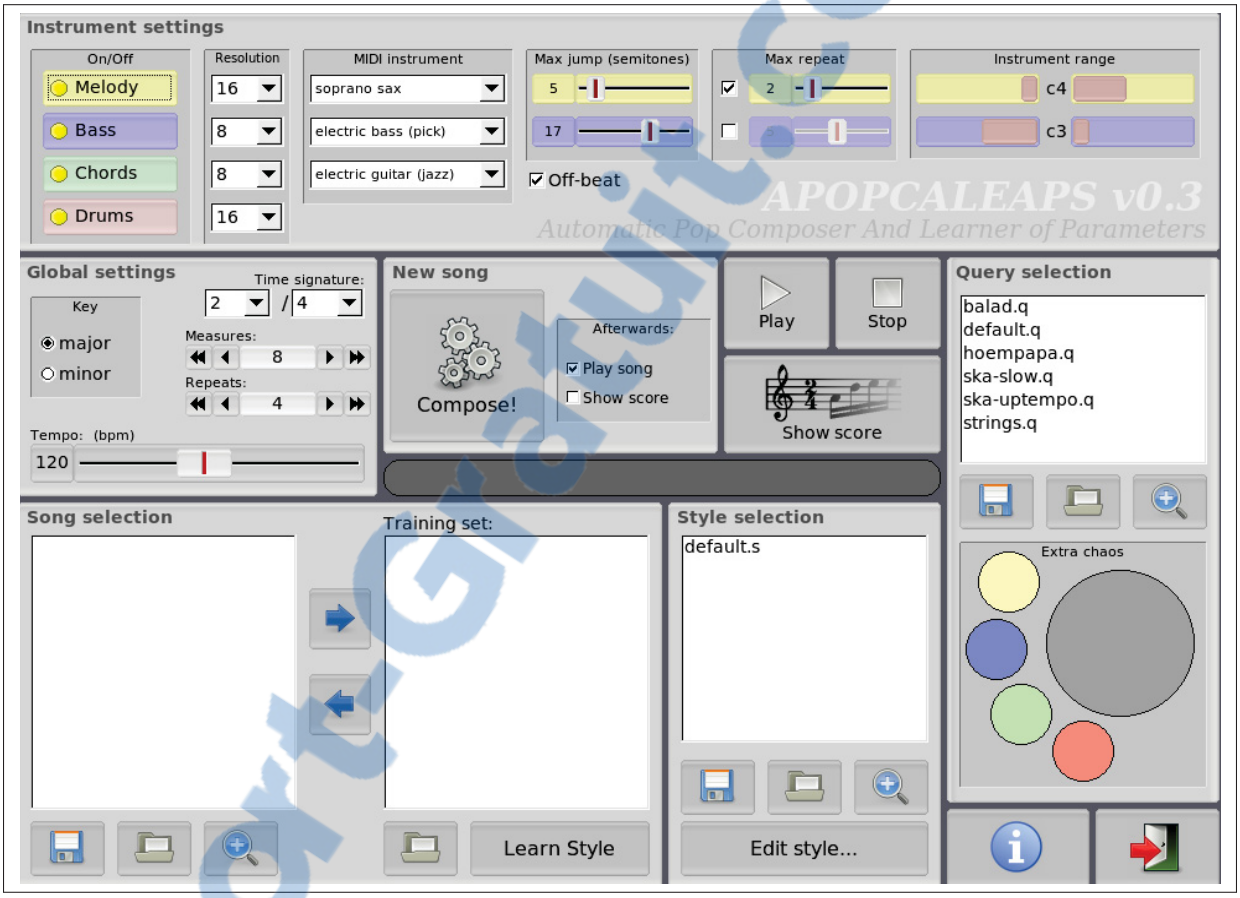

Figure 1.7 L'interface du prototype APOPCALEAPS
Tiré de (Sneyers et De Schreye, 2010)

SoudHelix de Schürger (2013) se présente comme un framework Java pour composer et jouer de la musique aléatoire dont la génération est basée sur des contraintes. Ce projet open source est un générateur de musique qui génère à partir d'un fichier XML de configuration des musiques unique au complet. Il ne permet pas de générer des mélodies. On doit écrire, dans un format particulier, quelques mesures de mélodies et d'accords pour différents instruments ainsi que différents paramètres concernant la structure du morceau, puis SoudHelix de charge de générer à partir de ses informations une pièce complète de plusieurs minutes au format MIDI. Il génère donc un arrangement d'un morceau à partir de la matière qu'on lui donne. Le prototype,

bien que jusqu'alors n'ayant pu être testé qu'en ligne de commande, offre des résultats limités, mais très prometteurs.

1.3 ZUI : Les interfaces utilisateur zoomables

> Big information worlds cause big problems for interfaces. There is too much to see. They are hard to navigate. (Furnas et Bederson, 1995)

Les interfaces utilisateur zoomables (ou ZUI pour Zoomable User Interface) représentent un champ précis de la recherche en interface graphique utilisateur (ou GUI pour Graphical User Interface). Une ZUI est une interface graphique sur laquelle l'utilisateur peut zoomer sur les objets affichés à l'écran. On navigue ainsi sur plusieurs niveaux d'abstraction. Chaque niveau d'abstraction offre l'affichage progressif des informations contenu dans les objets. Pad++, de Bederson et Hollan (1994), est un prototype ayant introduit les principales notions qui définissent les ZUI. Leur auteur ne définit pas Pad++ comme une application, mais comme un support pouvant accueillir des objets graphiques héritant des propriétés apportées par le système. Ces objets graphiques peuvent ainsi représenter n'importe qu'elle entité et cela a permis à différents sous-projets d'être développés autour de ce framework. Dans cet article, Bederson parle d'ailleurs d'un explorateur de fichiers qui a, par exemple, développé sur ce principe offrant une expérience nouvelle où l'utilisateur pouvait naviguer dans une arborescence de fichiers en zoomant et en se déplaçant sur le plan 2D.

C'est cette dernière expérimentation qui a sans doute inspiré les concepteurs de l'application commerciale Raskin (Raskin Software (2012)) dans laquelle l'utilisateur peut naviguer dans son arborescence de dossiers jusque dans les sous-dossiers et visualiser un aperçu du contenu des fichiers qu'ils contiennent en changeant le niveau de zoom et en se déplaçant. Sur la Figure 1.8, la vue principale montre une arborescence de fichiers dans laquelle on peut zoomer et dézoomer. L'amplitude du zoom est infinie, on peut partir d'un niveau de zoom où l'on peut voir l'ensemble des fichiers présents sur le disque, jusqu'à un niveau où l'on peut lire le contenu d'un fichier.

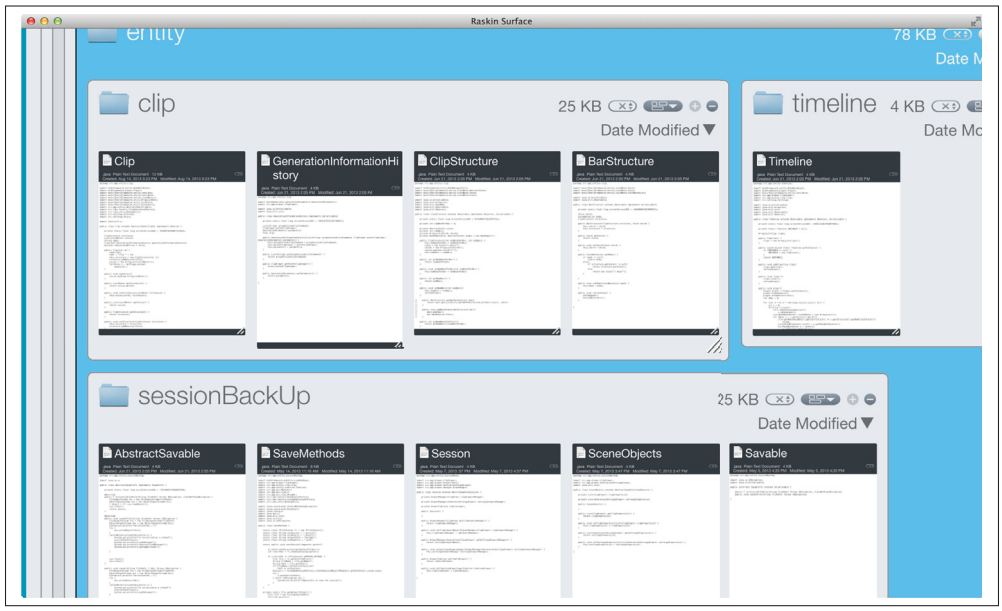

Figure 1.8 L'interface de Raskin
Imprime-écran de (Raskin Software, 2012)

Cette application est une bonne illustration de la métaphore du bureau. Ce dernier concept est poussé assez loin par Agarawala et Balakrishnan (2006) avec le prototype BumpTop (qui n'est pas une interface zoomable) en représentant un bureau physique en trois dimensions, vues du dessus, sur lequel sont représentés les fichiers présents sur disque sous la forme d'objets répartis en piles en fonction de leur type (photos, pages web, PDF, etc...). Cette dernière étude décrit une première évaluation au retour des utilisateurs très encourageants.

Pour ce qui est des algorithmes permettant d'implémenter des fonctions de zoom, Furnas et Bederson (1995) et Van Wijk et Nuij (2003) proposent des solutions. Nous avons d'ailleurs implémenté l'algorithme proposé par Furnas et Bederson (1995) dans notre prototype et il a entièrement répondu à nos besoins (voir 3.2.1). Van Wijk et Nuij (2003) est un travail plus récent qui cherche à améliorer l'expérience utilisateur en proposant un modèle plus complexe permettant de passer d'une vue zoomée à une autre vue zoomée en dézoomant de la vue courante, puis en faisant un panoramique, et pour finir en zoomant sur la nouvelle vue, le tout avec une animation fluide.

CHAPITRE 2

LA PROBLÉMATIQUE

Précédemment, des chercheurs ont développé des prototypes de générateurs de mélodies tels que le Melody Generator de Povel (2010) mais ces travaux se concentrent le plus souvent sur la conception, l'implémentation et les évaluations des algorithmes de génération. Ils négligent ainsi l'interface et on ne peut leur reprocher puisque leurs travaux n'ont pas vocation à mettre l'accent sur l'interaction utilisateur/générateur. Il nous semble qu'une interface plus conviviale pourrait permettre de toucher un plus large public et d'en augmenter ainsi l'impact. Une clé de l'intégration de ces algorithmes dans un processus de composition plus global est dans la manière de visualiser et manipuler les paramètres ainsi que les résultats. D'autres chercheurs ont créé des interfaces destinées aux compositeurs débutants comme le logiciel Hyperscore de Morwaread *et al.* (2007). Hyperscore permet de créer rapidement des pièces de musique sans connaissances préalables à partir de mélodies et de patrons rythmiques préexistants. Cette interface est d'ailleurs zoomable. Mais il n'y a pas de système permettant de zoomer automatiquement sur un objet sélectionné ou d'accéder à des vues différentes. L'aspect zoomable permet simplement de faire apparaitre exceptionnellement quelques informations supplémentaires sur le contenu des objets. Et Hyperscore n'intègre pas de générateurs de mélodies.

Nous avons voulu amener une réelle réflexion sur l'interfaçage utilisateur/générateur permettant d'utiliser des générateurs de mélodies pour soutenir une partie du processus de création. Aucun éditeur n'a, à l'heure actuelle, proposé de solution accessible pour le plus grand nombre. Nous savons cependant que certains y travaillent. C'est sans doute un enjeu dans le développement d'une nouvelle génération d'applications d'aide à la composition. Et les réponses ne semblent pas évidentes.

Si nous décidons d'implémenter un (ou des) algorithme(s) de génération, quel type devons nous choisir parmi les nombreuses approches (chaines de Markov, basé sur les règles, algorithmes génétiques) ? Au début de ce projet, nous avons essayé d'implémenter un algorithme basé sur les chaines de Markov et, après les lectures, nos conclusions sont que les chaines de Markov

ou n-grammes, lorsqu'elles sont utilisées pour générer des mélodies, sont adaptées pour écrire automatiquement dans le style d'un compositeur, d'un improvisateur ou d'en genre précis de musique, dépendamment du corpus choisi comme base de données. Mais ce que nous voulons c'est générer uniquement sous l'influence de paramètres pour que l'utilisateur puisse avoir la sensation d'avoir un certain contrôle sur le résultat est ainsi construire son style, même si le style des mélodies restera en grande partie déterminé par l'algorithme. Pour cela, la solution de développer un algorithme de génération basé sur les règles, dans le genre de celui de Povel (2010) semble être la plus efficiente. L'avantage de développer notre propre algorithme est d'avoir la possibilité d'adapter sa structure au besoin de notre démonstration pour évoluer avec notre interface. Cet algorithme est là pour soutenir le développement de notre interface, qui est le centre de nos préoccupations. Si les concepts intégrés à l'interface évoluent, par exemple si l'on pense qu'il peut être intéressant pour l'utilisateur de pouvoir générer un clip par la mutation de plusieurs clips existants, on peut modifier l'algorithme de génération pour qu'il puisse prendre ces passages musicaux en entrée et les utiliser. Si on veut rendre notre interface flexible, il nous faut une flexibilité des algorithmes de génération que l'on utilise qui nécessite que l'on ait un contrôle total sur la conception et le code de ces algorithmes.

La problématique est la suivante : de quelle manière une interface utilisateur peut-elle permettre l'utilisation de générateurs de mélodies, la visualisation et l'édition des paramètres de génération et des résultats des générations, ainsi que la visualisation de l'historique des paramètres utilisés pour générer les mélodies ?

Plusieurs questions se dessinent autour de cette problématique, révélant un peu plus le sens de ce travail : si nous démarrons le travail de conception en imaginant qu'une interface zoomable permette d'atteindre nos objectifs, quels concepts pouvons-nous développer et quelles fonctionnalités pouvons-nous intégrer à une interface zoomable afin de visualiser et éditer les paramètres de génération ainsi que les résultats des algorithmes de génération ? Pouvons-nous rendre l'utilisation de certains algorithmes de génération accessible aux utilisateurs novices en musique ou en informatique ? Sinon à quel groupe d'utilisateurs se destinerait une telle approche ?

Pour nous rapprocher de la solution, nous avons développé une application de création musicale intégrant des générateurs de mélodies. L'interface devait rendre l'utilisation de générateurs accessible pour les novices, ou du moins demander le moins de connaissances possibles, mais également pertinentes pour les utilisateurs intermédiaires et avancés. Un enjeu était d'atteindre notre cœur de cible, les utilisateurs débutants et intermédiaires, tout en conservant une puissance paramétrique et une flexibilité suffisante à satisfaire la volonté de compositeurs plus expérimentés. Il a fallu trouver des solutions de visualisation pertinentes permettant la manipulation des paramètres de génération et des passages musicaux générés, autrement dit des intrants et des extrants des algorithmes de génération. L'interface devrait être adaptée à l'utilisation d'algorithmes de génération basés sur les règles et l'aléatoire, pouvoir réutiliser la matière générée dans de nouvelles générations et de pouvoir visualiser l'historique des générations passées. Dans l'idée de rendre l'expérience utilisateur la plus agréable possible, l'utilisation devra tendre à être fluide et intuitive.

Cette succincte analyse nous amène à dresser la liste des exigences fonctionnelles suivantes pour le prototype :

1. Deux niveaux d'abstraction :
    (a) Le premier niveau d'abstraction permet la manipulation des objects et des liens qu'il y a entre eux, la visualisation de l'historique des générations et l'exécution de nouvelles générations. On a également un aperçu du contenu des objects représentant les paramètres de génération et de ceux représentant les résultats.
    (b) Le second niveau donne accès aux fonctionnalités d'édition des objets.
        i. Les options d'édition des objets représentant les paramètres de génération seront suffisantes pour traduire les paramètres de génération des algorithmes, définissant les règles et contraintes pour la génération de la hauteur des notes et du rythme.
        ii. Les fonctionnalités d'éditions des objets représentant les résultats des générations devront être suffisantes pour ajouter/supprimer des notes, visualiser si le

clip a été généré avec les bons paramètres harmoniques et rythmiques, pouvoir (re) définir la progression d'accord et les gammes.

(c) Les animations entre les niveaux de zoom se feront par une animation fluide et rapide.

2. Méthode de génération basée sur les règles :

(a) Les règles choisies pour l'harmonie et le rythme devront être sélectionnées pour être peu nombreuses et changer la valeur de leurs paramètres devra avoir le maximum d'effet sur les résultats des générations.

(b) Certains paramètres devront pouvoir changer dans le temps.

(c) Le générateur pourra prendre en entrée des passages, précédemment écrits ou générés, à partir desquels il pourra réutiliser des informations, garder certaines notes ou appliquer des mutations.

CHAPITRE 3

LE PROTOTYPE : GENSESSION

GenSession est une application Java qui utilise l'API javax.sound.

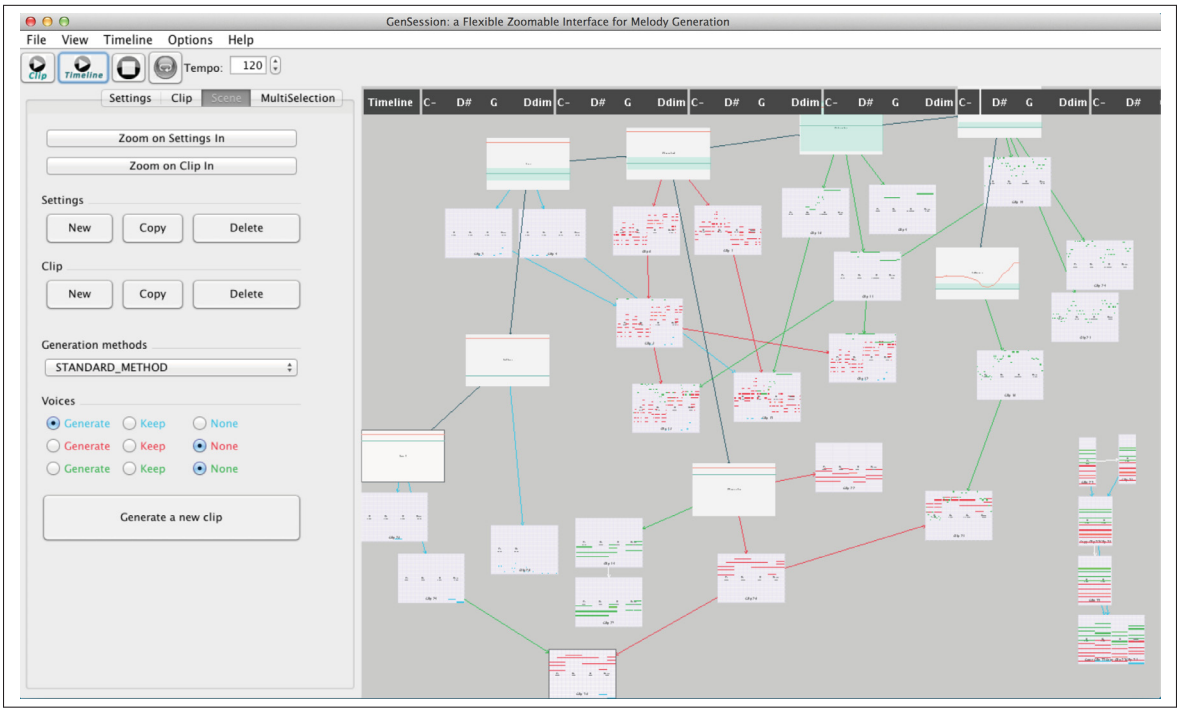

Figure 3.1 Fenêtre principale de GenSession avec vue dézoomée sur la scène
Imprime-écran de GenSession

3.1 Le survol général

GenSession permet à l'utilisateur de générer de courtes séquences musicales, appelées clips, utilisant différents paramètres de génération, pour ensuite les organiser sur un espace de travail 2D, les copier, les modifier, et supprimer ceux qu'il ne souhaite pas conserver. Dans la fenêtre principale (Figure 3.1), un panel d'options apparait dans le menu de gauche et la plus grande partie est prise par la scène, l'espace de travail, lequel contient des objets connectés les uns aux autres. Ces objets peuvent être vus comme des nœuds. Les liens entre eux sont représentés par des flèches qui indiquent quel nœud a fourni de l'information pour générer tels nœuds. Ces

nœuds sont donc des objets et il en existe deux types différents : les objets paramètres de génération représentent les données d'entrée pour les algorithmes de génération tandis que les clips représentent les séquences mélodiques générées et permettent leur édition. Les flèches entre les nœuds permettent à l'utilisateur de visualiser l'historique des opérations qui ont été exécutées telles que les générations de $clips$, les copies d'objets, les fusions de $clips$, en montrant les liens de parenté entre ces nœuds.

La Figure 3.1 montre le résultat d'une session de création d'une séquence musicale de 1 min 20 s pour trois instruments : basse (bleu), piano (rouge), violon (vert). Chaque couleur représente une voix et un canal MIDI.

3.2 L'interface zoomable

La possibilité de naviguer entre les différentes vues par l'intermédiaire d'une animation zoomant et dézoomant sur les différents objets est une caractéristique clé de notre interface. Ce zoom automatique est une animation qui fait à la fois un zoom et un panoramique. Il existe cependant une deuxième façon de faire un zoom, c'est le zoom manuel.

3.2.1 Le zoom automatique

L'algorithme permettant le zoom et le dé-zoom automatique entre les vues est issu des équations suivantes tirées de Furnas et Bederson (1995) :

$$z = \frac{z_1 - m z_1 x_1}{1 - mx} \tag{3.1}$$

où

$$m = \frac{z_2 - z_1}{z_2 x_2 - z_1 x_1} \tag{3.2}$$

Les équations 3.1 et 3.2 résolvent le problème que Furnas et Bederson (1995) nomme le « joint pan-zoom problem ». Il s'agit, lorsque l'on fait un zoom, de ne pas seulement changer le coefficient de zoom progressivement, mais aussi de faire se déplacer la vue sur le plan cartésien

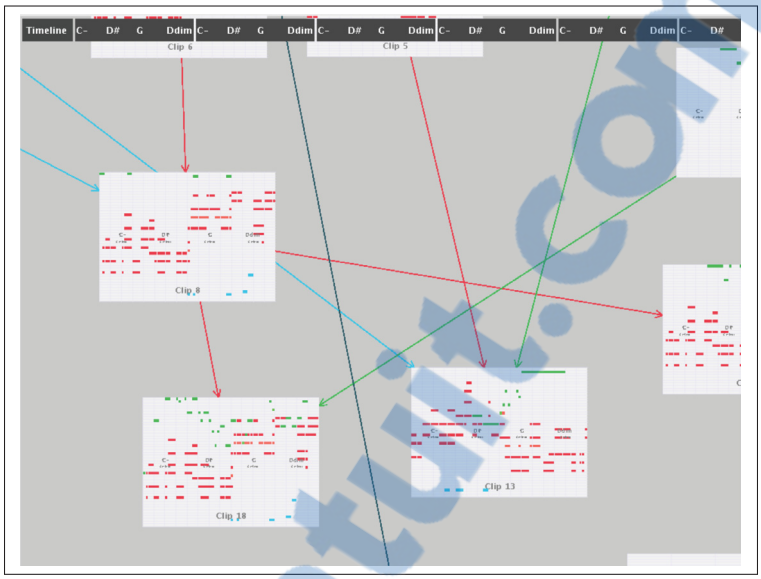

Figure 3.2 Zoom manuel sur une scène
Imprime-écran de GenSession

pour zoomer sur un point précis de façon fluide comme si on se dirigeait vers lui. Cette idée est expliquée par la Figure 3.3.

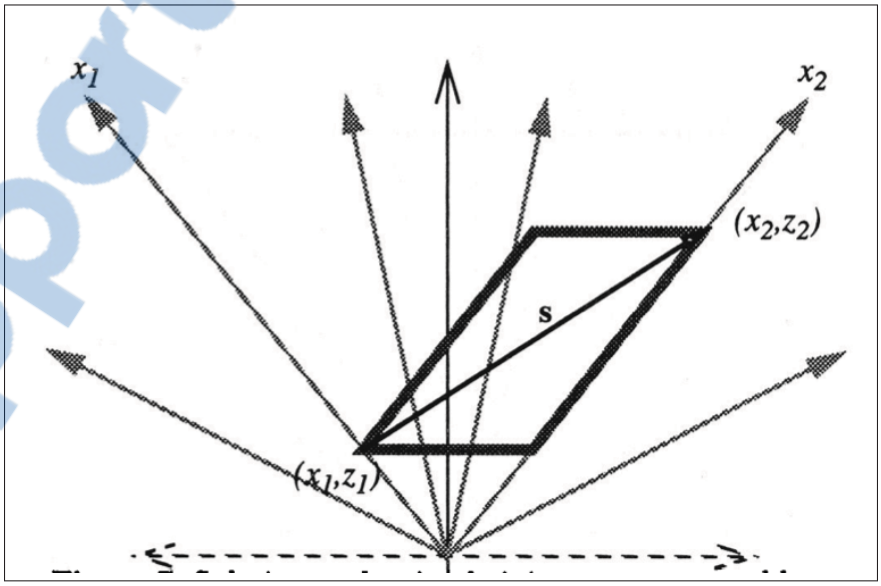

Figure 3.3 Solution proposée par Furnas pour son problème de panoramique et de zoom synchronisé lors d'un zoom sur un point
Tiré de (Furnas et Bederson, 1995)

3.2.2 Le zoom et panoramique manuel

Le zoom manuel fonctionne en maintenant le raccourci clavier Ctl et le $clic\ gauche$ enfoncé et en glissant la souris vers le haut ou vers le bas pour zoomer/dézoomer. Le panoramique manuel fonctionne en maintenant le raccourci clavier $SHIFT$ et le $clic\ gauche$ enfoncé et en glissant la souris dans la direction souhaitée pour le déplacement.

Cette fonctionnalité est valable sur toutes les vues. Lorsque le zoom ou le panoramique manuel est utilisé, le re-encadrement automatique est désactivé. La réactivation du re-encadrement automatique désactive l'effet du zoom et du panoramique manuel, pour revenir à la position par défaut de la vue.

3.2.3 Le ré-encadrement automatique

Appelé « re-framing » dans le prototype, cette fonctionnalité permet de donner l'illusion d'une scène infinie, en vue dézoomée, qui s'élargit automatiquement au fur et à mesure que l'on crée des objets ou qu'on les déplace vers l'extérieur de la scène. Inversement si l'on enlève des objets, la scène sera ré-encadrée de façon à ce que l'on voie toujours tous les clips. Lorsque l'on est zoomé sur un objet, ce mode s'assure que l'objet est centré et visible dans son ensemble.

3.3 Les voix

La notion de $voix$, dans le prototype, peut avoir plusieurs sens. Une $voix$ peut faire référence à la voix d'un instrument. Dans le prototype chaque voix peut être lu au format MIDI et les messages MIDI peuvent être redirigés vers n'importe quel port MIDI. Ainsi n'importe quel séquenceur ou instrument virtuel peut lire les messages d'une voix. On pourrait aussi exporter une ou plusieurs voix dans un fichier MIDI, importé ce fichier MIDI dans un logiciel d'édition de partition telle que Finale (MakeMusic (1988-2012)), et faire jouer la partition écrite dans une notation standard à un musicien. Une $voix$ peut donc être jouée par un ou des musiciens réels.

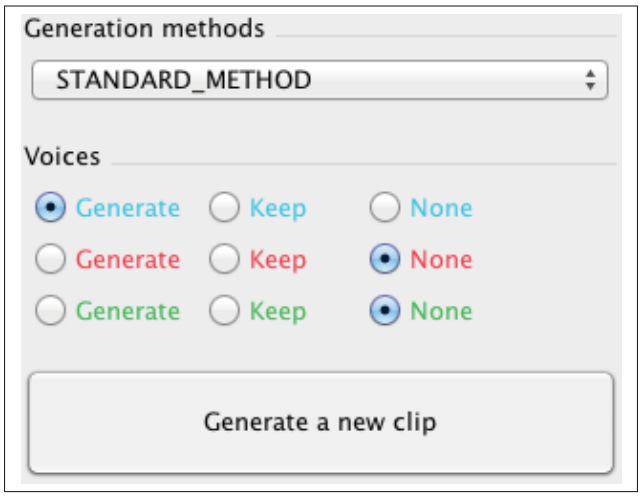

Figure 3.4 Menu de gauche en vue dézoomée
Imprime-écran de GenSession

Sur la Figure 3.2, on peut distinguer des *clips* possédant 3 voix. Chacune des voix est de couleur différente. Ils héritent ces voix d'autres clips ayant préalablement été générés.

La Figure 3.4 montre que l'on peut choisir sur quelles voix on souhaite générer la ou les nouvelles mélodies et que l'on peut aussi garder des voix provenant d'un *clip* pré-existant.

Par défaut, chaque *voix* est attribuée à un canal MIDI, tel que la première voix est attribuée au canal 1, la seconde au canal 2, et ainsi de suite. Si on lit un clip qui possède une séquence musicale écrite/générée sur plusieurs voix, chaque voix pourra alors être récupérée dans un canal MIDI différent. Ensuite, les messages de chaque canal MIDI pourront être redigirés vers un instrument virtuel différent. C'est de cette façon que chacune des voix peut être jouée par un instrument différent. Une *voix* représente donc plus concrètement un cannal MIDI ou plutôt l'ensemble des notes attribuées à un canal.

3.4 Les objets paramètres de génération

Quand l'application s'exécute pour la première fois, la scène est vide, il n'y a aucun nœud. L'utilisateur peut commencer par créer un clip vide et y écrire des notes manuellement. Il pourrait d'ailleurs créer plusieurs clips, utiliser les fonctionnalités d'éditions, les mettre à la

suite les uns des autres et ainsi créer une séquence musicale sans utiliser les fonctionnalités de génération. Bien que cela soit permis, l'application est conçu pour un autre genre de processus de création et celui-ci est dirigé par la génération (de séquences musicales). Ainsi, dans le scénario le plus commun, l'utilisateur commence par créer un objet paramètres de génération. Une fois créé il peut zoomer sur l'objet pour l'éditer et modifier les paramètres.

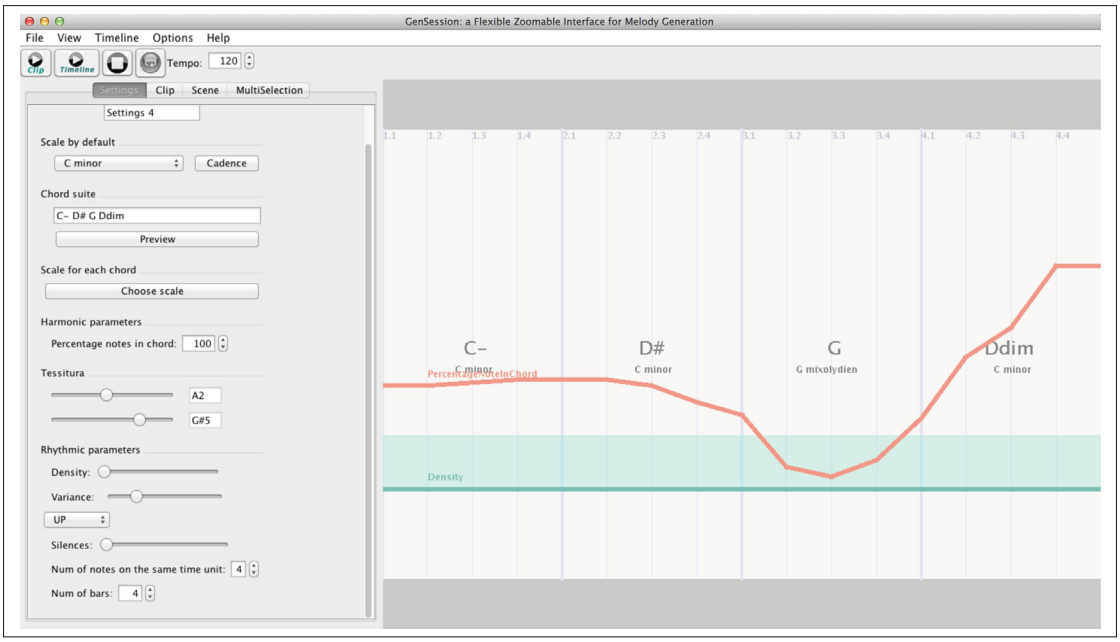

Figure 3.5 La vue d'édition d'un objet *paramètres de génération* après que l'utilisateur ait zoomé sur cet objet
Imprime-écran de GenSession

Les objets paramètres de génération sont donc utilisés pour générer des clips et contiennent les paramètres conditionnant en partie la matière musicale générer. Quand l'utilisateur a zoomé sur un objet paramètres de génération, celui-ci prend tout l'espace de la scène et le panneau d'options sur la gauche de la fenêtre principale permet à l'utilisateur de modifier les gammes, la progression d'accord, le nombre de mesures des clips postérieurement générés ainsi que bien d'autres paramètres.

La Figure 3.5 montre la vue d'édition d'un objet *paramètres de génération* après que l'utilisateur aie zoomé dessus. On peut voir les paramètres globaux définis dans le menu de gauche

et les paramètres dynamiques représentés par les deux courbes de couleurs rouge et verte. Les paramètres y sont configurés pour générer des clips de 4 mesures, une gamme de « Do mineur (C minor) pour les mesures 1, 2 et 4, le mode de Sol (G) mixolidien pour la mesure 3 et une progression d'accord Do mineur — Re# majeur — Sol majeur — Re diminué » (ou « C— D# G Ddim » en écriture internationale) avec un accord par mesure.

3.4.1 Paramètre global et paramètre dynamique

Nous avons fait la distinction entre deux types de paramètres. Les *paramètres globaux*, globaux et les *paramètres dynamiques*. Les paramètres dynamiques peuvent aussi être considérés comme *globaux*, mais dans ce cas leur valeur n'évolue pas dans le temps et ils perdent leur dynamisme. Les *paramètres dynamiques* peuvent ainsi varier temporellement dans les objets paramètres de génération en étant directement dessinés sur l'objet en pressant une touche du clavier. Les valeurs sont échantillonnées au temps ainsi une valeur est enregistrée pour chaque temps de chaque mesure. Si une valeur est entrée pour le paramètre dans le menu d'option à gauche, la valeur s'appliquera pour toute la séquence et le paramètre devient ainsi *global*. Par exemple dans la Figure 3.5, le pourcentage des notes dans l'accord démarre autour de 50%, diminue à la mesure 3 et tend vers les 75% à la fin de la mesure 4.

3.4.2 Les paramètres de génération

Les objets paramètres de génération contiennent donc des informations représentant des paramètres qui seront plus tard utilisés comme données en entrée lors d'exécution d'un algorithme de génération. Les paramètres pouvant être modifiés lorsque l'utilisateur a zoomé sur objet paramètres de génération sont les suivants :

La progression d'accords représente les accords cibles et peut être défini en entrant directement le nom des accords dans le champ texte sur la gauche ou en définissant la cadence correspondante en chiffre romain d'après la convention : ici « I V II I » pour Do mineur. Il est à noter que lorsque l'on génère un clip, les accords cibles dans la progression d'accords ne

sont pas simplement copiés dans le clip, mais représentent un paramètre harmonique ayant une grande influence sur le résultat d'une génération.

Le pourcentage de note dans l'accord représente de pourcentage de notes générées dont la hauteur est choisie parmi les notes qui constituent l'accord cible de la mesure courante. Si la hauteur de la note n'est pas choisie dans l'accord, elle est choisie parmi les notes qui constituent la gamme de la mesure courante durant la génération.

La densité rythmique, c'est la durée moyenne d'une note. Elle peut varier en fonction de la variance. Plus celle-ci est élevée, plus l'algorithme a la liberté de choisir des durées éloignées de celle configurée. La densité rythmique varie entre 0 et 6, et détermine la durée de chaque note générée sauf dans le cas où celle-ci est déterminée par un autre critère comme la durée des notes d'un clip préexistant. Nous avons choisi le mapping suivant : une densité de 0 correspond à une ronde, 1 à une blanche, 2 à une noire pointée, 3 à une noire, 4 à une croche pointée, 5 à une croche, 6 à une double croche. L'utilisateur peut aussi configurer une variance qui est elle aussi un paramètre dynamique.

La variance de la densité définit la liberté laissée à l'algorithme de génération de choisir une densité rythmique inférieure ou supérieure à la densité définie par l'utilisateur. Par exemple si la densité rythmique, dessinée dans un objet paramètres de génération , est de 3 avec une variance à 1 pour un instant t, la densité rythmique résultante à cet instant t de la séquence générée sera choisie aléatoirement dans l'intervalle 3 ± 1 et déterminera la durée de la note correspondante.

Directivité de la variance de la densité : La variance a une directivité. Cette directivité peut prendre trois valeurs : «up», «down» ou «both». C'est-à-dire que la densité peut varier au-dessus, en dessous ou les deux en fonction de cette directivité. Par exemple, la variance est de 2 et la directivité est «up», la densité ne pourra qu'être égale ou supérieure à celle définit par l'utilisateur, autrement dit elle pourra varier de +1 ou +2 seulement.

Le pourcentage de silence, c'est le pourcentage de chance pour chaque note d'être un silence ou une note jouée.

Le nombre de notes sur la même unité temporelle est simplement le nombre de notes K générées simultanément sur une même voix. Ce nombre est un entier naturel inférieur ou égal à 8. Si K=1, alors une seule et unique note sera générée sur chaque unité temporelle où le générateur aura choisi de mettre au moins une note. Si K>1 alors on obtiendra des accords. Nous avons fait le choix de limiter ce nombre à 8, car si la valeur est trop importante les résultats nous ont semblé moins pertinents.

La tessiture : Chaque instrument, réel ou virtuel, est soumis à des contraintes physiques ou techniques, qui lui imposent une tessiture. C'est-à-dire le spectre sonore de l'instrument, délimité par la note la plus aiguë et la note la plus grave qu'il peut émettre. Ici la $tessiture$, en tant que paramètre de génération, définit l'ensemble des hauteurs de note autorisées par l'intervalle entre la note la plus haute et la note la plus basse pouvant être écrite par l'algorithme. On peut donc choisir la tessiture en fonction des limites de l'instrument pour lequel on souhaite générer la séquence. Mais on peut aussi définir cet intervalle en fonction du registre dans lequel on souhaite générer la mélodie. Le registre est un intervalle de notes compris dans la tessiture d'un instrument. En effet, un instrument tel que la guitare possède une tessiture large, allant d'une note relativement aiguë à une note relativement grave, couvrant plus de 3 octaves. On pourrait ainsi choisir de ne générer que sur le registre grave de la guitare définit alors par un intervalle de notes plus restreint que celui de sa tessiture. La tessiture la plus large possible définissable en paramètre est celle imposée par les limites du MIDI. En MIDI les hauteurs des notes sont codées de 0 à 127. La note la plus haute est un Sol de la 7^{e}octave et la note la plus basse est un Do de l'octave -2, selon la convention.

3.5 Les objets clips

Le second type de nœud sur la scène est un type d'objet appelé $clip$. Le $clip$ contient une séquence musicale écrite par un humain ou un algorithme. Sur la scène le clip est représenté par une vue miniature du pianoroll, le nom des accords sur chaque mesure, son nom apparait en dessous. Quand l'utilisateur zoom dans un clip, il peut voir une partition de la séquence musicale écrite sous forme de pianoroll. Chaque voix est d'une couleur différente. Le nombre de

voix doit être déterminé dans le code avant compilation. L'évaluation du prototype a été effectuée avec 2 voix, mais des tests concluants ont été effectués avec 3 voix comme en témoigne la Figure 3.6. Par défaut, la première voix est bleue, la seconde est rouge et s'il y en a une troisième, elle est verte. L'utilisateur peut éditer les voix et ainsi ajouter et supprimer des notes en cliquant sur le pianoroll. Il doit sélectionner d'avance la voix qu'il souhaite éditer dans le menu d'option à gauche. Les notes les plus aiguës et les plus graves n'apparaissent pas directement et l'utilisateur peut faire défiler vers le haut ou vers le bas pour les faire apparaitre en maintenant pressée une touche de clavier ainsi que le clic droit de la souris. L'utilisateur peut aussi, à l'aide d'un autre raccourci et de la même façon, déplacer le pianoroll sur l'axe des abscisses.

Figure 3.6 Un objet *clip* en vue zoom
Imprime-écran de GenSession

Dans la vue d'édition d'un clip, comme sur la Figure 3.6, ont peut voir sur le pianoroll des lignes grisées qui surlignent les notes faisant partie de l'accord de chaque mesure. Cette fonctionnalité peut être désactivée ou bien activée pour mettre en surveillance d'autres informations comme les notes qui sont dans la gamme courante, pour chaque mesure, ou les notes consti-

tuant les enrichissements des accords comme la 7e, la 9e ou encore la 11e. Ces outils peuvent aider l'utilisateur à évaluer les résultats d'une génération ou bien à éditer plus rapidement en visualisant la fonction des notes dans l'harmonie choisie en fonction de leur placement temporel dans la séquence.

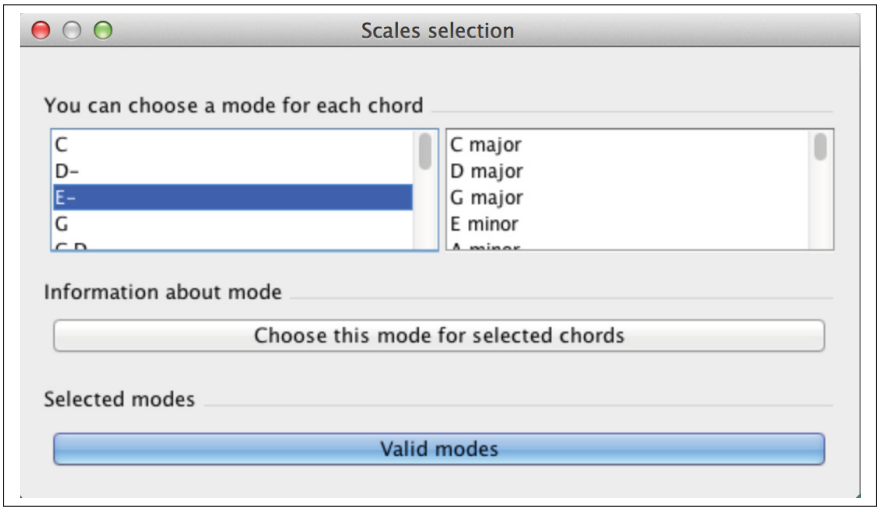

Figure 3.7 Boite de dialogue qui permet d'associer une gamme différente de la gamme par défaut à chaque accord
Imprime-écran de GenSession

3.5.1 Les options des clips

Le menu de gauche donne accès aux fonctionnalités suivantes, visibles sur la Figure 3.6 de haut en bas :

Editer le nom du clip qui apparaît sur la vue miniature du clip lorsque l'on est dézoomé. Permet aussi de repérer le clip sur la timeline ou après l'avoir exporté. Le nom par défaut est « Clip » + un numéro de clip (ex :« Clip » 18 sur la Figure 3.2).

Activer/désactiver la fonction Re-Generation qui lorsqu'elle est activée entraine la re génération du clip, à chaque fois qu'on en lance la lecture, à partir des mêmes informations que pour les générations précédentes. Si le clip est sur la timeline, la re génération s'exécute aussi.

Choix de la gamme par défaut parmi une liste de gammes possible en fonction des notes déjà écrites. Si aucune gamme ne contient l'ensemble des notes de la séquence alors la liste complète des gammes est affichée dans le menu déroulant.

Bouton cadence qui ouvre une boite de dialogue permettant de rentrer une progression d'accord dans le champ «Suite d'accords» à partir d'une cadence représentée par un enchainement de degrés écrit en chiffre romain (par ex : I III V II donne, pour la gamme de do mineur, les accords Do mineur, Eb majeur, Sol majeur, et Re diminué).

Le champ texte «Suite d'accord» est un champ texte où doit être écrit une progression d'accord, les accords sont écrits selon la notation jazz standard. Chaque accord est assigné à une mesure.

Le bouton «preview» ouvre une boite de dialogue qui permet d'écouter les accords de la progression d'accord.

Activer/désactiver le bouton «Note relative aux accords et à l'harmonie» qui contrôle la transposition intelligente décrite plus loin (section 3.11.2)

Le bouton «choix des gammes» ouvre une boite de dialogue qui permet de choisir une gamme différente de la gamme par défaut pour les mesures assignées à certains accords. Ainsi si une gamme est sélectionnée dans ce menu pour un accord (parmi la liste des gammes qui contiennent les notes de cet accord), les mesures sont assignées à cet accord, seront assignées à cette gamme. Voir Figure 3.7.

Le nombre de mesures correspond au nombre de mesures du clip.

Le boutons radio «Voix» permet de sélectionner la voix que l'on souhaite éditer. Le nom de chaque voix est écrit dans sa couleur qui permet d'en identifier les notes sur le clip.

Le boutons radio «Durée de note» permet de sélectionner la durée de note courante d'édition sur le pianoroll. Ainsi si l'on ajoute une note sur le pianoroll, cette note aura la durée sélectionnée. Les durées sont représentées par les figures de note correspondantes. (Choix possibles : Ronde 𝅝 — Blanche 𝅗𝅥 — Noire ♩ — Croche ♪ — Double croche 𝅘𝅥𝅯).

Les interrupteurs à bascule «Surlignage des notes» sont au nombre de trois. Ils permettent d'activer/désactiver le surlignage de certaines notes sur le pianoroll pour faciliter la lecture et l'édition. Le premier permet de surligner les notes contenues dans l'accord de chaque mesure, le second permet de surligner les notes contenues dans les enrichissements des accords de chaque mesure (s'il y des enrichissements), le troisième permet de surligner les notes de la gamme de chaque mesure.

3.5.2 La sélection multiple

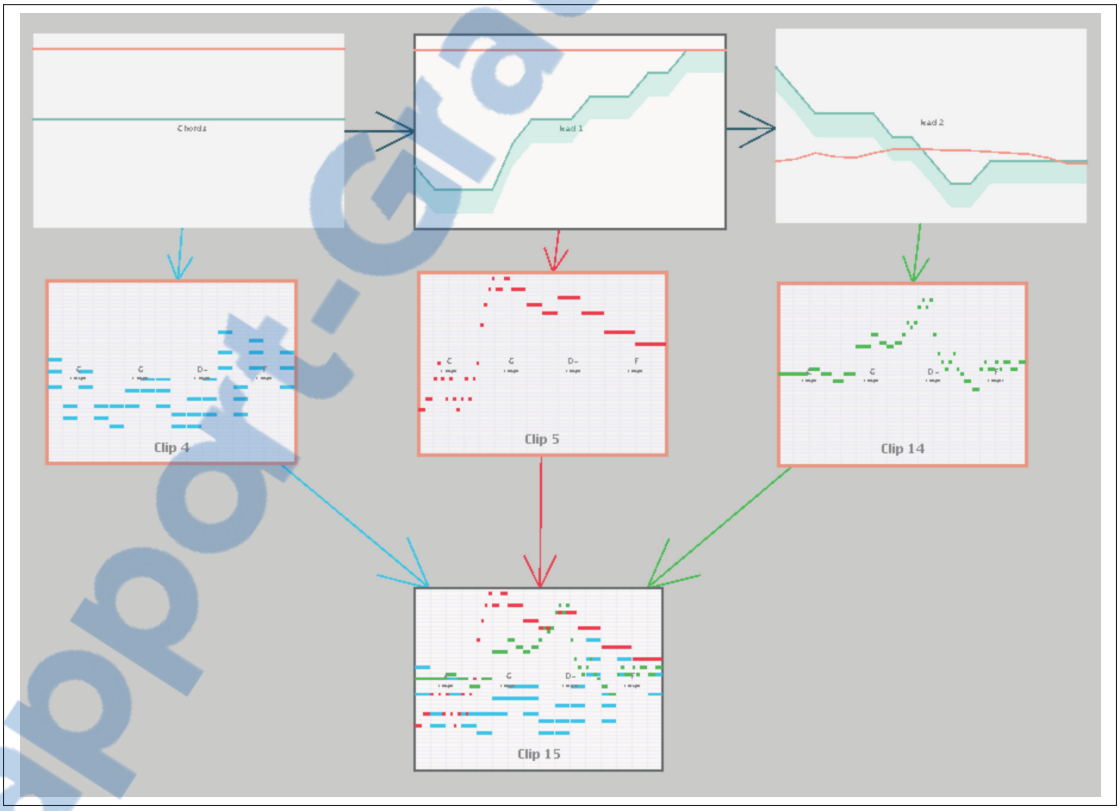

Figure 3.8 Exemple de fusion des voix
Imprime-écran de GenSession

La sélection multiple permet à la fois de fusionner les voix de plusieurs *clips* dans un nouveau *clip* (illustrer par la Figure 3.8), mais aussi de coller deux *clips* à la suite (illustrer par la Figure 3.9).

Pour faire apparaitre le menu de sélection multiple dans le panneau de gauche il faut maintenir la touche $SHIFT$ enfoncée et cliquer sur les clips que l'on souhaite fusionner les voix.

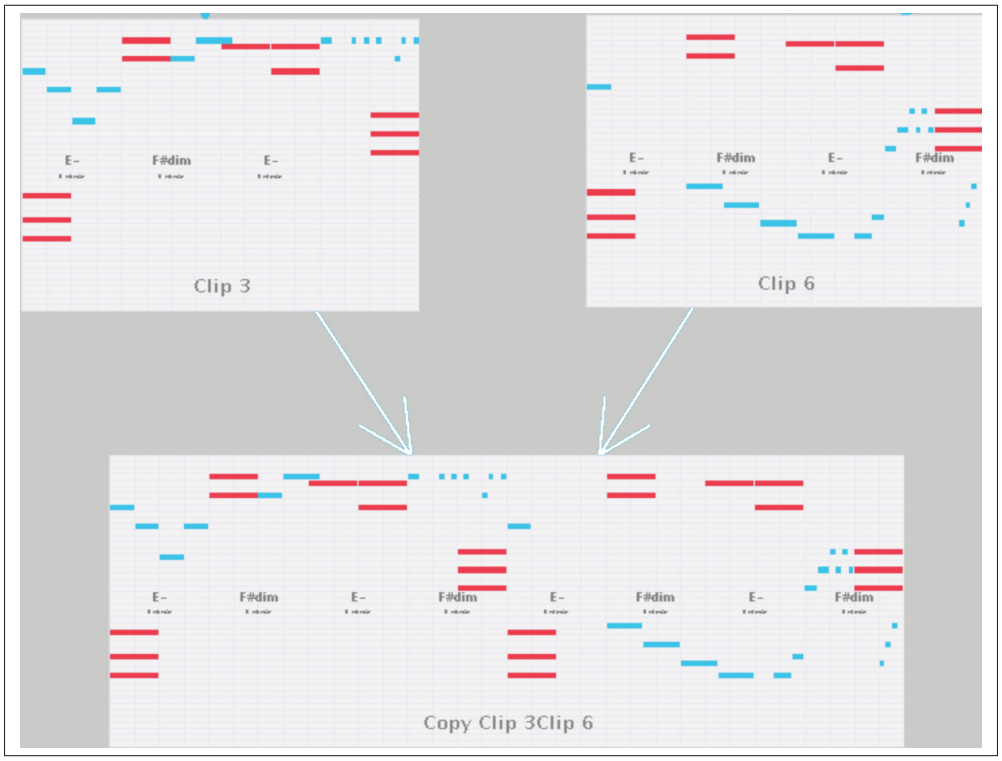

Figure 3.9 Deux clips collés l'un à la suite de l'autre
Imprime-écran de GenSession

Ensuite, deux fonctionnalités s'offrent à nous :

La fusion des clips : La Figure 3.8 montre que, après avoir généré plusieurs mélodies sur les voix différentes de plusieurs clips, on peut fusionner les différents clips en gardant les voix désirées de chacun de ces clips. Elle montre un clip, avec 3 voix différentes, généré indirectement à partir de 3 paramètres de génération. Pour ce faire, il faut faire apparaitre le menu de gauche comme sur la Figure 3.10. On peut sélectionner pour quel clip, parmi les clips sélectionnés, on souhaite extraire chacune des voix. Les voix sont ensuite réunies dans le nouveau clip. Le nouveau clip fils est lié aux clips dont il hérite les voix.

Coller des clips : La Figure 3.9 montre un exemple de deux clips collés les uns à la suite des autres dans un nouveau clip enfant héritant de ses deux parents.

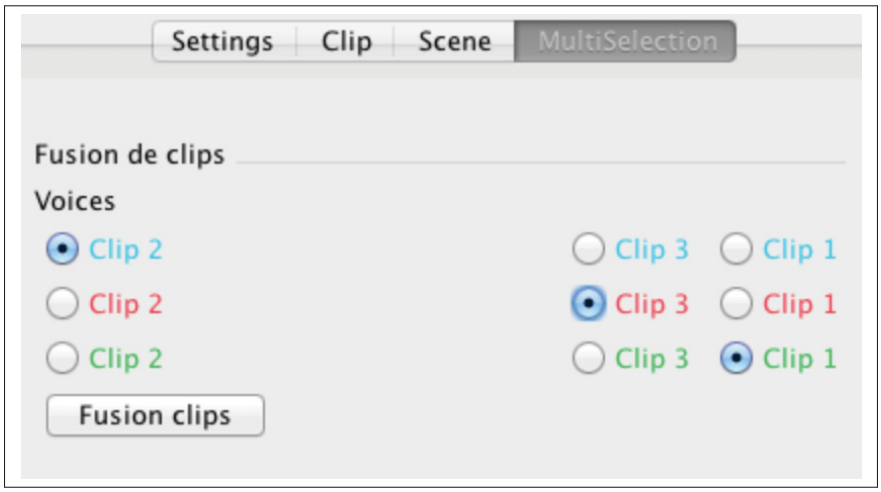

Figure 3.10 Le menu de sélection multiple
Imprime-écran de GenSession

3.6 La timeline

La timeline est représentée en haut de la scène. Elle permet de mettre les clips les uns à la suite des autres sur une ligne de temps similaire dans l'idée à celles que l'on peut trouver dans la plupart des logiciels d'animation.

Les clips peuvent être :

– glissés-déposés à la dernière position ;

– insérés entre deux clips précédemment déposés ;

– ou bien encore, remplacer un clip déjà présent.

Si l'on passe la souris sur le rectangle représentant un clip sur la timeline, l'objet clip correspondant est encadré sur la scène. Sur la Figure 3.12, on peut voir que sur les rectangles représentant les clips apparaissent les progressions d'accords.

Figure 3.11 La timeline lorsqu'elle est vide.
Imprime-écran de GenSession.

Figure 3.12 Un exemple de timeline avec deux clips séquencés
Imprime-écran de GenSession

La timeline peut être lue par le bouton «lecture» marqué «timeline» dans la barre de transport.

3.7 La lecture des clips et de la timeline

Figure 3.13 La barre de transport est affichée dans le panneau du haut de la fenêtre principale
Imprime-écran de GenSession

La barre d'outils du panneau en haut à gauche de l'interface de GenSession est appelée $barre\ de\ transport$ (voir la Figure 3.13). Elle comprend 4 boutons :

- Le bouton $lecture\ du\ clip$ qui lance la lecture du clip sélectionné sur la scène ou sur lequel on est zoomé.
- Le bouton $lecture\ de\ la\ timeline$ qui va lire à la suite les clips placés sur la timeline.
- Le bouton $stop$ qui arrête la lecture en cours. Si aucune lecture n'est en cours, il n'a pas d'effet.
- Le bouton à deux états *répéter la lecture*. S'il est enfoncé, la lecture du $clip$ ou de la $timeline$ se fera en boucle jusqu'à que l'on l'arrête manuellement avec le bouton stop, en relançant une autre lecture ou en changeant à nouveau l'état de ce bouton.

On peut également changer le tempo auquel les séquences MIDI seront lus en changeant la valeur du composant de type «spinner» à droite de la *barre de transport*. Par défaut cette valeur est à 120 BPM (battements par minute).

La classe Java représentant le lecteur MIDI utilise l'API Java.Sound pour envoyer les messages sur le bon port et le bon canal MIDI.

3.8 Le système de sauvegarde et d'export

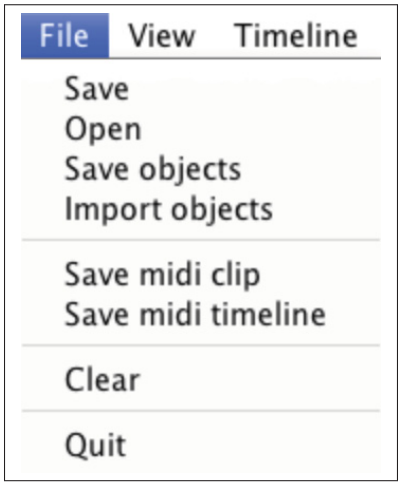

Figure 3.14 L'onglet *File* du menu d'option
Imprime-écran de GenSession

L'onglet *File* du menu d'options, montré sur la Figure 3.14, donne accès à plusieurs fonctionnalités d'import et d'export des objets. Elles sont détaillées ci-dessous.

Save permet de sauvegarder toute la *session*. C'est-à-dire tous les objets qui sont sur la scène et les liens qu'il y a entre eux.

Open permet d'ouvrir une *session*. Cela permet de charger tous les objets précédemment enregistrés sur une nouvelle scène.

Save objects permet de sauvegarder les objects sélectionnés avec l'outil de sélection multiple, ainsi que les liens qu'il y a entre eux, dans un fichier.

Import objects permet de charger sur la scène les objects précédemment enregistrer avec la méthode $save\ objects$, ainsi que les liens qu'il y a entre eux.

Save MIDI clip permet d'exporté le $clip$ sélectionné dans un fichier MIDI.

Save MIDI timeline permet de sauvegarder les $clips$ placés sur la $timeline$ dans un fichier MIDI.

Clear permet de vider la *scène* de tous les objets présents. Cela revient à créer une nouvelle *scène* vide en effaçant la $session$ courante.

3.9 Les raccourcis

Les raccourcis sont importants pour l'ergonomie. Bien que les fonctionnalités sont accessibles avec la souris, les raccourcis clavier permettent de gagner du temps.

- Voici ceux qui permettent de gérer le zoom :

  Barre espace La $barre\ espace$ permet de lancer le zoom sur un objet clip ou bien dé-zoomé depuis n'importe qu'elle vue.

  La touche s La touche s permet de lancer le zoom sur un objet paramètres de génération ou bien dézoomé depuis n'importe qu'elle vue.

  SHIFT + clic Maintenir $SHIFT\ +\ clic$ enfoncé permet de faire un panoramique en déplaçant la souris sur les axes XY.

  Ctl + clic Maintenir $Ctl\ +\ clic$ enfoncé permet de faire un zoom/dé-zoom en déplaçant la souris sur l'axe Y.

- Ceux actifs lorsque l'on est zoomé sur un objet paramètres de génération :

  D + clic Maintenir enfoncée $D\ +\ clic$ pour dessiner la courbe de densité rythmique.

  F + clic Maintenir enfoncée $F\ +\ clic$ pour dessiner la courbe de pourcentage de note dans l'accord.

- Ceux actifs lorsque l'on est zoomé sur un objet clip :

  Clt + Shift + clic Maintenir enfoncé $Ctl + Shift$, puis cliquer sur le pianoroll pour faire jouer une note.

3.10 L'algorithme de génération

> Music is multidimensional. When listening to music, we perceive various aspects such as rhythm, melody, harmony, or instrumentation simultaneously. (Anders *et al.*, 2003)

Pour générer de nouveaux clips, l'utilisateur doit être en vue dézoomée comme sur la Figure 3.1 auquel cas le panneau d'options à gauche contient des éléments qui permettent de sélectionner une méthode de génération et d'exécuter un algorithme avec en entrée les paramètres extraits de l'objet paramètres de génération sélectionnée et éventuellement du clip sélectionné. La Figure 3.4) montre les options correspondantes de ce panneau de gauche. Le bouton « Generate a new clip » permet de lancer une nouvelle génération. On voit aussi que l'on peut choisir sur quelles voix on souhaite générer des notes à partir de l'objet paramètres de génération sélectionnée, et quelles voix on souhaite conserver du clip sélectionné (s'il y a lieu).

L'algorithme de génération développé pour le prototype se divise en deux parties : la première permet de générer la hauteur des notes, il s'agit de l'algorithme de génération harmonique alors que la seconde qui génère les rythmes est l'algorithme de génération rythmique. La Figure 3.15 montre le diagramme de classe représentant l'architecture du module de génération du prototype. Ce module est dépendant du module de théorie musicale (voir section 3.11).

Nous avons donc dit que notre algorithme, utilisé par défaut, génère le rythme et la hauteur des notes des nouvelles séquences. Or, nous avons aussi implémenté des variantes de cet algorithme qui récupère des informations contenues dans les clips existants. Nous en avons développé 3 variantes.

Nous allons dans un premier temps voir comment fonctionne la méthode par défaut, puis nous détaillerons comment fonctionne l'algorithme de génération harmonique et l'algorithme de

génération rythmique. Par la suite, nous ferons une description de chacune des 3 variantes de génération.

3.10.1 La méthode de génération par défaut

L'algorithme de génération dont nous parlons est inspiré en partie par le travail de Povel (2010). Dans son prototype «Melody Generator», l'algorithme de génération est stochastique. Il est basé sur des règles d'harmonies, mais le résultat n'est pas seulement déterminé par elle et l'aléatoire (ou pseudo-aléatoire) a une part importante.

Si l'on regarde le processus à haut niveau, l'algorithme que nous avons implémenté génère les notes séquentiellement. Une note commence sur l'unité temporelle suivante de celle où se termine la note ou le silence précédent (en dehors de la première note qui commence sur la première unité temporelle du clip). La position de la nouvelle note (ou du silence) est égale à la position de la précédente plus sa durée. La durée de la nouvelle note dépend de la densité rythmique à l'instant de départ de cette nouvelle note définie dans les paramètres. Si cette densité rythmique, pour cette unité temporelle, a une variance > 0 alors un choix stochastique (aléatoire, mais dépendant de la densité) définit la nouvelle durée. Quand ces paramètres de densité et de variance l'autorisent, l'algorithme choisira le plus souvent la même durée que la note qui précède. Ce $taux\ de\ variation$ de la densité est défini en tant que constante dans le code, mais pourrait devenir un paramètre de génération si le prototype évoluait.

Un des paramètres de génération est appelé dans le prototype «num of notes on the same time unit» (voir section 3.4.2) et est ce que nous pourrions appeler, avec plus d'élégance, le $coefficient\ polyphonique$. Il s'agit du nombre de notes à générer sur chaque position temporelle pour une seule voix. Par exemple si ce coefficient est de 1 alors une note (ou silence) sera écrite sur chaque unité de temps. Cela signifie que selon la logique séquentielle décrite plus haut, seulement une seule note devrait être écrite à chaque fois. Ce paramètre est défini de manière globale, mais il serait intéressant de le rendre dynamique ou le faire varier aléatoirement dans le temps (voir section 3.4.1).

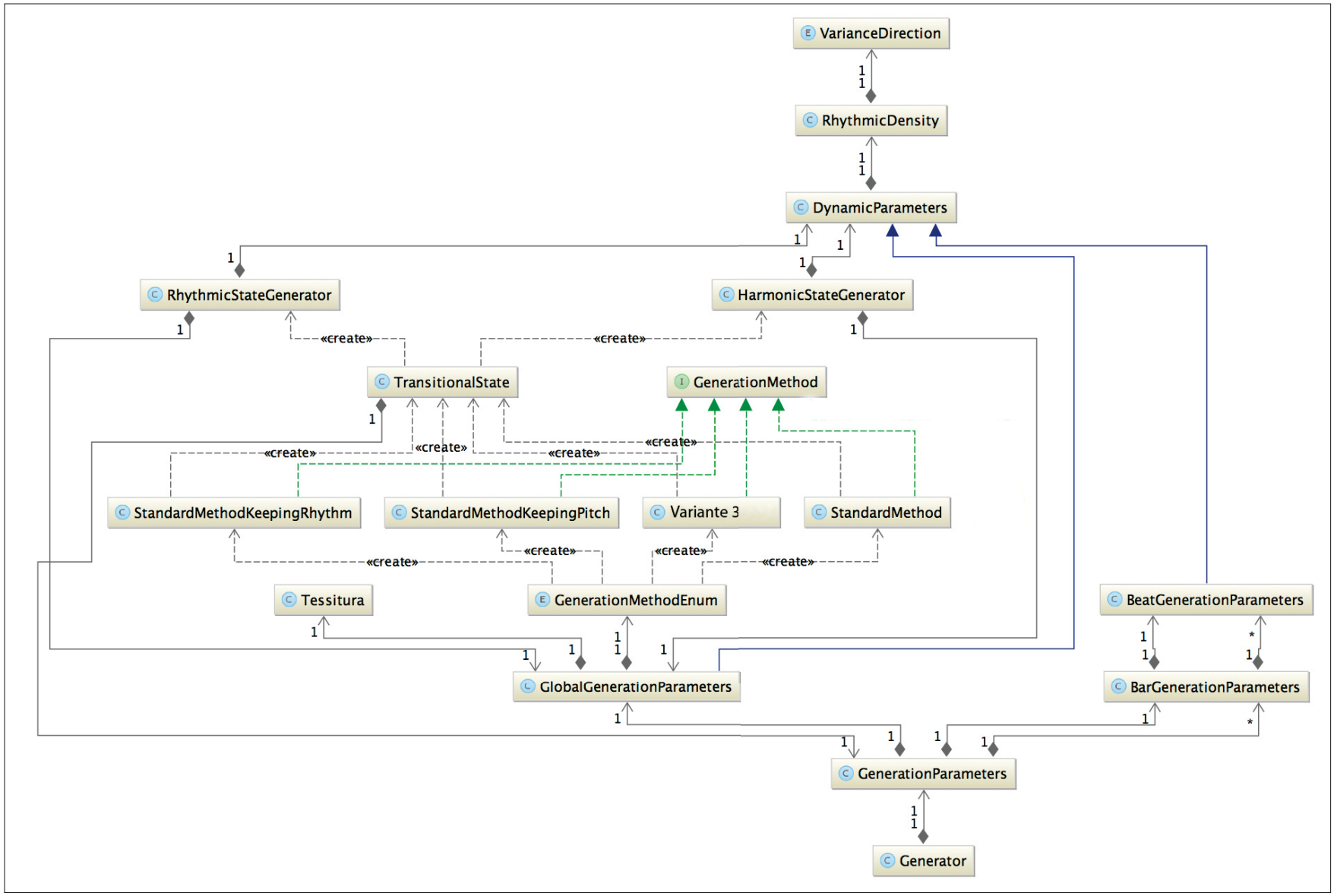

Figure 3.15 Diagramme de classe représentant l'architecture du module de génération du prototype
Ce diagramme a été généré avec l'outil UML de l'IDE Intellij

Définissons K le nombre de notes à générer sur la même position temporelle représentant ce $coefficient\ polyphonique$. À chaque fois qu'une note est générée (K — 1) autres notes sont aussi générées avec la même durée et la même position dans le temps, mais une hauteur différente. On obtient alors des accords. Par exemple, K = 1 signifie qu'une mélodie monophonique est générée, tandis que K = 3 signifie que des accords de 3 notes seront générés. Si le «pourcentage de note dans l'accord» est configuré à 100% alors les accords générés seront des renversements des accords cibles. Cela signifie qu'ils seront construits des sons des accords cibles, mais dans un ordre potentiellement différent de leur état fondamental. On dit qu'un accord est renversé lorsque sa basse n'est pas le fondamental. Si par exemple nous prenons l'accord de Do majeur. Il est constitué des 3 sons suivant Do Mi Sol. La basse de l'état fondamental est Do. S'il est joué, sur un instrument autorisant la polyphonie, sa note la plus grave doit être un Do. Mais dans notre cas, les renversements sont autorisés, ce qui signifie que la note la plus grave jouée peut être indifféremment un Do, un Mi ou un Sol. Nous pourrions ainsi noter la séquence de notes générées comme $(n_{1,1}, \ldots, n_{1,K}), (n_{2,1}, \ldots, n_{2,K}), \ldots$, ou $(n_{t,1}, \ldots, n_{t,K})$ représentant l'ensemble des notes écrites pour être jouées à chaque position temporelle t.

L'algorithme crée, pour exécuter la génération d'une nouvelle note ou d'un nouvel accord, une instance de la classe « TransitionalState ». Son constructeur prend en paramètre les paramètres de génération et la note précédente s'il y a lieu. Ensuite l'*état transitoire* exécute l'algorithme de génération rythmique pour générer la position de la note dans le temps (dont il déduit les paramètres dynamiques définit pour cette position temporelle), ainsi que la durée de la note. La note rythmique de type $RhythmicNote$ (voir à la section 3.11.1 à propos des entités) retourné par cet algorithme est passée en paramètre de l'algorithme de génération harmonique qui détermine la hauteur de la note et renvoie un objet de type Key. L'objet $Note$ renvoyé par l'*état transitoire* est constitué de la note rythmique $RhythmicNote$ et de la hauteur Key. Pour générer un accord (qui n'est autre qu'un ensemble de plusieurs notes avec les mêmes informations rythmiques), l'état transitoire génère premièrement la note rythmique, puis ensuite plusieurs hauteurs de notes différentes, pour finalement renvoyer une liste de notes $Note$.

3.10.2 L'algorithme de génération rythmique

Comme le montre le diagramme de la Figure 3.15, la classe du générateur chargée d'assigner une position dans le temps et une durée à une nouvelle note est notée «RhythmicStateGenerator» et sa méthode $createRhythmicNote$ est appelée par une instance de l'état transitoire «TransitionalState».

L'algorithme commence par déterminer la position de départ de la note. S'il n'y a pas de note précédente passée en paramètre, la position de départ sera la première unité de temps de la nouvelle séquence. Si par contre il y a une note qui précède, la nouvelle note commencera sur l'unité de temps suivante de celle où finit la précédente.

Une fois la position de la note définie, l'algorithme peut déterminer les paramètres dynamiques concernant le rythme, dont la densité rythmique, applicable.

L'algorithme va ensuite déterminer la durée de la note par un choix aléatoire sous la contrainte des paramètres. Les densités minimum et maximum autorisées sont déterminées en fonction de la densité, de la variance et de la directivité de la variance définie par l'utilisateur. Un choix aléatoire est ensuite fait pour choisir une densité comprise entre la densité minimum et maximum. Une fois la densité choisie elle est traduite par la durée à laquelle elle correspond. Puis la nouvelle note rythmique générée est retournée constituée de la position de départ de la note et de sa durée.

3.10.3 L'algorithme de génération harmonique

Si l'on se réfère au diagramme de la Figure 3.15, la classe du générateur chargée de générer les hauteurs de notes en fonction de l'harmonie est notée, «HarmonicStateGenerator» et sa méthode $newPitch$ sont appelées par une instance de l'état transitoire «TransitionalState». $newPitch$ prend en paramètre les paramètres de génération, la note précédente s'il y a lieu, ainsi que l'instance de « RhythmicNote » représentant les informations rythmiques précédemment générées par l'algorithme de génération rythmique.

Choisissons p le pourcentage de notes qui doit être dans les accords cibles. Ce pourcentage est défini par l'utilisateur dans les objets paramètres de génération.

La première note, $(n_{1,1})$ est donnée par une hauteur aléatoire, avec la probabilité p d'être choisie parmi les sons de l'accord cible de la première mesure, et la probabilité p-1 d'être choisie parmi les degrés de la gamme. À partir de $(n_{t,1})$ nous générons $(n_{t,2})$ et ainsi de suite jusqu'à la génération de $(n_{t,k})$ auquel point $(n_{t,1})$ est utilisé pour générer $(n_{t+1,1})$, laquelle est utilisée pour générer $(n_{t+1,2})$, ...

À chaque fois, qu'une note est générée, elle a 50% de chance d'être plus aiguë et 50% de chance d'être plus grave. Une fois ce choix effectué, l'algorithme vérifie qu'une hauteur plus haute ou plus basse répond bien aux contraintes imposées par la $tessiture$. Si la note sort de l'intervalle autorisé, alors une nouvelle hauteur est choisie aléatoirement, mais cette fois en prenant en compte la contrainte : elle sera dans la direction opposée à celle précédemment choisie (si elle était plus aiguë que la précédente, elle sera plus grave ; et inversement dans le cas contraire). Ainsi si la note était plus haute que la précédente et si elle était en dehors de la tessiture, cette fois-ci elle sera plus basse.

3.10.4 La variante 1 : gardant le rythme des notes

Le principe est simple : garder le rythme et re-générer la hauteur des notes. Autrement dit, on souhaite garder les positions dans le temps et les durées des notes de la séquence de notes d'un clip existant et en changer uniquement les hauteurs pour obtenir une nouvelle séquence dans un nouveau clip. Ces nouvelles hauteurs sont générées avec des paramètres harmoniques définis dans un objet paramètres de génération. Le nouveau clip généré est ainsi apparenté à deux nœuds : un objet paramètres de génération et un objet clip.

En pratique, le cas d'utilisation standard est le suivant. L'utilisateur a préalablement crée sur la scène un objet clip, ici d'une génération ou non, contenant une séquence de notes. Il a également créé et configuré un objet paramètre de génération. Il sélectionne le clip voulu ainsi que l'objet paramètres de génération par un simple clip, sélectionne la bonne méthode de généra-

Figure 3.16 Un clip généré avec la variante 1, gardant le rythme d'un clip préexistant
Imprime-écran de GenSession

tion «keeping rhythm» dans le menu de droite ainsi que la ou les voix sur lesquelles il souhaite générer, puis clic sur le bouton «generate a new clip» dans ce même menu. Le nouveau clip apparaît donc sur la scène en dessous de l'objet paramètres de génération. La Figure 3.16 nous montre un exemple de clip généré avec la variante 1. On peut voir les deux liens de parenté entre le clip fils et les deux nœuds parents. Ces deux nœuds parents sont le clip dont il hérite le rythme et l'objet paramètres de génération dont il hérite les paramètres harmoniques).

Les paramètres rythmiques de l'objet paramètres de génération, tel que la densité rythmique, ne nous intéressent pas ici. S'ils ont été modifiés tant mieux, car ça pourrait servir plus tard, mais cela n'aura pas d'impact lors de la génération avec cette variante et seuls les paramètres harmoniques seront pris en compte par le générateur. Le rythme de la nouvelle séquence est déterminé par le rythme du clip sélectionné.

3.10.5 La variante 2 : gardant la hauteur des notes

Cette variante est similaire, dans la logique, à la variante 1. Dans celle-ci, on garde la hauteur des notes du clip sélectionné et on génère le rythme à partir du paramètre de génération. Dire qu'on garde la hauteur des notes, c'est juste, mais notre algorithme va un peu plus loin que ça. Si une hauteur de note change de mesure et que la mesure dans laquelle se situe cette nouvelle note n'a pas la même harmonie que la précédente, alors la fonction de la note dans l'ancienne harmonie sera conservée et la nouvelle note sera transposée dans sa nouvelle harmonie (voir la transposition intelligente à la section 3.11.2).

3.10.6 La variante 3 : mélangeant aléatoirement les autres variantes

Cette variante réutilise les algorithmes des variantes 1 et 2 ainsi que ceux de la méthode par défaut. Cet algorithme fait appel aux algorithmes décrits plus haut pour générer la nouvelle séquence. Il itère sur les notes de la séquence du clip sélectionné et pour chacune, exécute aléatoirement l'une des «mutations» suivantes :

- La note d'origine est gardée telle qu'elle, ses informations rythmiques et de hauteurs sont conservées.
- L'algorithme de la variante 1 est appliqué sur la note d'origine. Une nouvelle note est créée, partageant la même position dans le temps et la même durée que la note d'origine, mais avec une hauteur différente.
- L'algorithme de la variante 2 est appliqué sur la note d'origine. Une nouvelle note est créée ayant la même fonction dans l'harmonie (relative à sa hauteur) que la note d'origine, même position de départ, mais avec une durée potentiellement différente.
- L'algorithme par défaut est appliqué. Une nouvelle note est alors générée en fonction de la note précédente de la nouvelle séquence s'il y a lieu, mais sans prendre en compte la note de la séquence d'origine.

La nouvelle séquence du nouveau clip généré est ensuite retournée.

3.11 Le module de gestion de la théorie musicale

Le module de théorie musicale est la ressource principale des algorithmes du module de génération. Il est également abondamment utilisé par le module principal qui dessine l'interface. Il permet de faire traduire les notes, manipulées dans le programme sous la forme de l'entité $Note$, vers une entité $MidiNoteEvent$ du module qui gère le MIDI et en permet la lecture.

3.11.1 Codage des informations : les entités

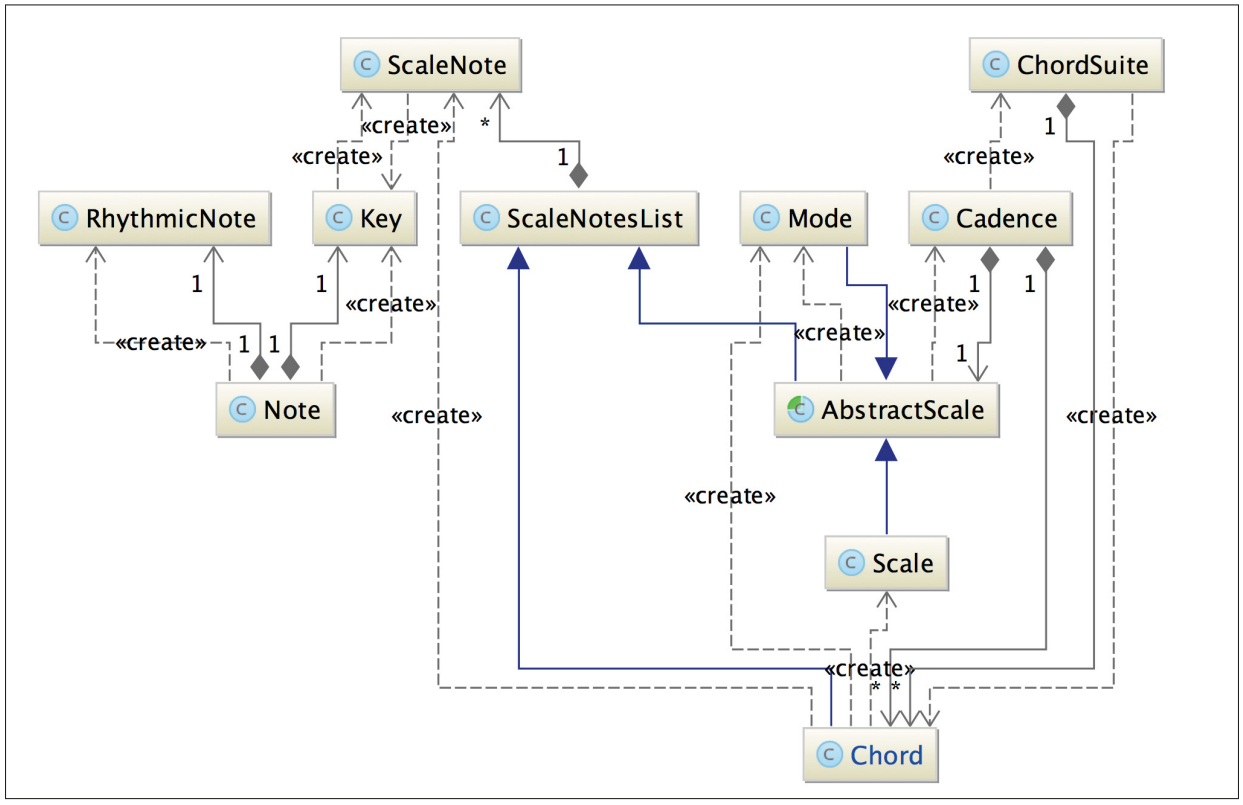

Figure 3.17 Diagramme de classe représentant l'architecture du module de théorie musicale du prototype
Ce diagramme a été généré avec l'outil UML de l'IDE Intellij

L'architecture du module de théorie musicale est représentée sur la figure 3.17 qui montre les liens entre ses entités. Ce module est en autre utilisé par le générateur de mélodie. En voici une description des principales entités :

Note Les notes sont codées sous forme d'un objet java $Note$ qui lui-même contient deux objets : la note rythmique $RhythmicNote$, et la hauteur appelée Key. Ainsi les informations rythmiques et harmoniques peuvent être dissociées.

ScaleNotesList Les classes représentant les gammes, les modes et les accords héritent de la classe $ScaleNoteList$ et sont donc des surcharges de listes d'objets $ScaleNote$ encapsulés.

ScaleNote La classe $ScaleNote$ représente des notes qui ont uniquement une hauteur indépendamment de toute octave et de toute information rythmique. Le terme «scale note» pourrait être traduit par «note d'échelle». Il s'agit ainsi d'une note destinée à constituer des gammes, modes ou accords sous la forme de liste. Ensuite, elles peuvent être transposées à n'importe qu'elle octave pour donner des notes de type Key.

RhythmicNote Les objets $RhythmicNote$ contiennent les deux informations rythmiques nécessaires à la note pour la traduire en MIDI et la jouer. Il s'agit de la position de départ de la note dans le temps ainsi que de sa durée.

3.11.2 La transposition intelligente

Conserver la fonction d'une note lors du changement de l'harmonie : c'est l'objectif de cette fonctionnalité bien utile pour modifier les accords ou les gammes des clips existant tout en gardant la séquence préalablement écrite ou générée en cohérence avec la nouvelle harmonie. Cet algorithme est aussi utilisé pour la variante 2 de notre algorithme de génération.

On peut aimer le comportement général d'une phrase musicale, son rythme et sa courbe mélodique par exemple, mais trouver que certaines notes dans l'harmonie sonnent mal et se demander quel aurait était le résultat si on avait généré ce clip avec des paramètres harmoniques différents tel qu'une progression d'accords différente. Cette fonctionnalité permet de faire cela. Alors que l'on change la progression d'accords ainsi que la ou les gammes associées à une séquence (dans un clip), on peut choisir de conserver la fonction de chaque note et ainsi d'obtenir la séquence (du clip) transposée en cohérence avec la nouvelle structure harmonique. Cette fonctionnalité est seulement accessible lorsque l'on est zoomé sur un clip. Pour l'activer ou

la désactiver, il suffit de cliquer, dans le menu d'options à gauche, sur le bouton à deux états «note relative to chords and scale» qui signifie «note relative aux accords et à la gamme». Toute modification des gammes ou accords lorsque la fonction est activée entrainera une transposition selon ce principe. Cela permet donc de tester de nouveaux paramètres harmoniques rapidement, mais c'est aussi une puissante fonctionnalité d'édition d'un clip existant. Le seul autre moyen d'essayer rapidement une progression d'accords différente sans changer les autres paramètres aurait été de copier l'objet paramètres de génération qui a permis de générer ce clip, modifier la suite d'accords, lancer une nouvelle génération et obtenir un nouveau clip avec une nouvelle phrase respectant la nouvelle harmonie. Mais avec cette dernière méthode, dont la procédure est plus fastidieuse, on perd la courbe mélodique.

L'algorithme 1 décrit le fonctionnement de la transposition intelligente en reprenant les noms de classe et les relations du diagramme 3.17. La notation de ce pseudo-code est proche du langage java dont il reprend la syntaxe de typage, les types primitifs et les opérateurs d'incrémentation.

L'algorithme fonctionne de la manière suivante. Il y a 5 entrées : la séquence avant le changement de gammes ou d'accords noté seq ; la suite d'accords avant notés $previousProgression$; la gamme associée à chaque accord avant changement noté $previousScales$; la progression d'accord associé après noté $nextProgression$; les gammes après notées $nextScales$. On récupère en sortie la séquence de notes transposées dans la nouvelle harmonie notée $nextSeq$.

Une limite de cet algorithme est qu'il ne prend pas en compte les notes qui sont en dehors des échelles harmoniques définies (gammes et accords) et qui sont alors ignorées. Ces notes peuvent être des notes de passages ajoutées par l'utilisateur. L'algorithme pourrait être amélioré pour les prendre en compte, mais il ne nous semblait pas urgent d'ajouter cette amélioration pour mener à bien notre expérience.

Algorithm 1 Transpose les notes de la séquence seq vers la nouvelle $nextSeq$

Require: Defined $seq, previousProgression, previousScales, nextProgression, nextScales$

List<Note> $seq \leftarrow clipSelected.getNotesSequence$

List<Note> $nextSeq \leftarrow new$ List<Note>()

for int $i = 0 \rightarrow (seq.lenght - 1)$ **do** ▶ for each note from the input sequence

Note $note \leftarrow seq.getNoteByIndex(i)$ ▶ get the number i note sequence

ScaleNote $scaleNote \leftarrow note.getkey.getScaleNote$

int $noteBarNum \leftarrow note.getRhythmicNote.getBar$

Chord $chord \leftarrow previousProgression.getChord(noteBarNum)$

ScaleNote $rootNote \leftarrow chord.getRootNote$

Mode $mode \leftarrow previousScales.getScale(chord).getMode(rootNote)$

int $notePositionInMode \leftarrow previousProgression.getScale.getNotePosition(scaleNote)$

Scale $nextScale \leftarrow nextProgression.getScale(noteBarNum)$

ScaleNote $nextRootNote \leftarrow nextProgression.getChord.getRootNote$

Mode $nextMode \leftarrow nextScale.getMode(nextRootNote)$

ScaleNote $nextScaleNote \leftarrow nextMode.getNoteFromPosition(notePositionInMode)$

Key $nextKey \leftarrow note.getkey.getCloserKey(nextScaleNote)$

Note $nextNote \leftarrow newNote(note.getRhythmicNote, nextKey)$

$nextSeq.add(nextNote)$

i++

end for

return $nextSeq$

Ensure: $nextSeq.lenght == seq.lenght$

CHAPITRE 4

L'ÉVALUATION

4.1 La rétroaction des utilisateurs

4.1.1 Le protocole

Pour évaluer notre prototype, nous avons procédé à des tests utilisateurs auprès de sujets humains. C'est tests nous ont permis de faire une première évaluation qualitative de GenSession.

La version compilée du prototype GenSession utilisé peut être télécharger sur la page web de présentation du projet : http://hifiv.ca/~francoiscabrol/GenSession/ (Cabrol (2013)). Le nombre de voix utilisables dans les clips étant décidé avant compilation, celui-ci ne peut pas changer au cours de l'exécution du programme. Dans cette version, les voix sont au nombre de 2. L'une est bleue et l'autre est rouge.

Pour cette étude, les utilisateurs sont classés en trois groupes de niveaux : les *débutants* qui ont une connaissance très limitée de la théorie musicale ; les *intermédiaires* qui ont une expérience significative de la pratique musicale et une maitrise des notions de base du solfège ; les *confirmés* qui ont une connaissance précisent du solfège et une bonne expérience en tant que musicien et/ou compositeur.

Les rencontres avec les utilisateurs se sont déroulées au cours de deux semaines du mois de mai 2013. Ces rencontres individuelles ont été conduites par nous-mêmes avec sept utilisateurs, chacun d'entre eux ayant un lien professionnel avec la musique. Trois d'entre eux sont des étudiants en master dans le domaine de la musique et technologie, un est étudiant en master en acoustique, deux sont des étudiants doctorants faisant de la recherche liée à la musique et technologie, et un autre est un artiste diplômé des beaux-arts ayant participé à des performances, devant un public, de musique électro-acoustique. Deux de ces utilisateurs sont des musiciens expérimentés (*confirmés*), trois d'entre eux sont des musiciens *intermédiaires*, tandis que les deux autres sont *débutants*.

La machine utilisée pour faire l'expérience était notre ordinateur portable personnel, le même ayant servi au développement. L'écran était de 13 pouces et une souris sans fil branchée en USB venait compléter le trackpad intégré pour le contrôle du curseur.

Avant de commencer, et souvent à leur demande, nous donnions une estimation du temps de la rencontre à environs 30-45min. Nous précisions cependant qu'il s'agissait du temps minimum que nous estimions nécessaire pour recueillir assez d'informations pour réaliser l'étude, et que s'il souhaitait continuer plus longtemps c'était bien entendu possible.

Nous commencions les rencontres par une petite démonstration des principales fonctionnalités du prototype pendant à peu près 15 minutes. Une présentation se déroulait typiquement selon le plan de scénario en annexe I et était la même pour chaque utilisateur. Dans les grandes lignes, après un tour rapide des méthodes de génération et des fonctionnalités d'éditions, elle montrait comment générer des clips avec une mélodie sur la voix bleue, puis avec une mélodie polyphonique (accords) sur la voix rouge. Ensuite des clips avec des accords sur la voix rouge étaient générés, puis ces clips étaient fusionnés avec les mélodies «bleues» des clips précédemment générés vers de nouveaux clips comme sur la figure 3.16.

Après cette présentation, les participants étaient invités à interagir avec le prototype. Si nous pressentions qu'ils avaient des questions, nous les encouragions à les poser afin de comprendre leurs difficultés, mais aussi afin de les voir progresser le plus rapidement possible. Il nous a semblé que meilleure était leur compréhension du fonctionnement du prototype, plus diverses étaient leurs interactions et leurs réactions.

Quand les utilisateurs commençaient à utiliser le prototype, ils démarraient avec la scène vide d'une nouvelle instance de GenSession. Ils étaient invités a créer ce qui leur plaisait, d'explorer au maximum de leurs possibilités et de tenter de faire ce qu'ils avaient envie. Nous restions à côté en tout temps pour répondre à leurs questions oralement, essayant de créer un maximum d'échanges.

4.1.2 Les résultats

Au cours des séances d'évaluation, nous prenions les notes sur papier. Puis, plus tard, nous retranscrivions ces notes dans un document texte. Les notes prises au cours de ces rencontres sont la principale source, avec nos souvenirs encore frais, dont sont extraites les informations constituantes de cette sous-partie. L'intégralité de ces notes est retranscrite en annexe II.

Voici ce que nous en retenons :

Après la présentation du prototype, nous laissions donc l'utilisateur prendre en main l'interface. Dans un premier temps, la plupart des utilisateurs reproduisent les étapes qui leur ont été montrées lors de la démonstration : créer un objet *paramètre de génération*, ensuite générer des clips, ensuite combiner différente voix de différents clips. Par contre, chaque utilisateur a choisi des paramètres de génération différents, des gammes et des progressions d'accords différentes et a joué avec les paramètres de densité rythmique.

Les utilisateurs ont, dans leur ensemble, beaucoup aimé l'interface utilisateur zoomable (ZUI), et ils ont aimé l'idée de représenter les objets (dont les clips) en miniature en vue dézoomée : ils ont trouvé que ça leur permettait de les distinguer entre eux aisément. Le mode de re-encadrement (re frame) automatique de la scène leur a aussi vraiment plu : un utilisateur a dit, en découvrant le principe, qu'il souhaiterait voir apparaitre une fonctionnalité similaire dans d'autres logiciels d'édition musicale qu'il utilise régulièrement.

Tous les utilisateurs, y compris les débutants trouvent l'utilisation du prototype accessible, mais ils trouvent aussi qu'il faut un temps d'apprentissage pour assimiler les fonctionnalités variées de l'interface utilisateur. Après 30-45 minutes d'utilisation, ils avaient pu tester toutes les principales fonctionnalités et se sont montrés intéressés pour l'utiliser plus longuement. Comme un utilisateur l'a précisé, « Good interface, easy to use, but requires a training time like every music production tool to understand all the functions. ».

Cependant, une étape s'est révélée critique et a posé de la difficulté à un certain nombre d'utilisateurs. Pour générer un premier clip, le choix de la gamme et de la progression d'accords

est complexe, tel qu'il est proposé, sans connaissance en théorie musicale. Le système d'harmonisation des gammes retrouvant les accords correspondant à une cadence (gamme et suite de degrés d'accords) permet de pallier un peu le problème et par son usage tous les utilisateurs peuvent passer l'étape. Les deux utilisateurs débutants ont trouvé tout de même la tâche complexe et ont eu peu de contrôle sur le choix des accords au travers de l'écriture d'une cadence et du choix de la gamme, concepts qui ne leur sont pas familiers.

Plusieurs utilisateurs ont aussi eu des difficultés à retenir les raccourcis clavier qui n'étaient, semble-t-il, pas toujours bien choisies. Par exemple le fait de zoomer/dézoomer sur un clip avec la touche $barre\ espace$ n'a pas été jugé intuitif. Heureusement cette fonctionnalité était aussi présente en tant que bouton dans le panneau d'option. Nous avons quand même dû à plusieurs reprises écrire la liste des principaux raccourcis sur un papier, les utilisateurs n'arrivant pas à les retenir.

Les cinq utilisateurs avec des expériences en musique de débutant et d'intermédiaire ont trouvé que le prototype autorisait un processus de création accessible produisant des séquences mélodiques sonnant bien, tandis que les utilisateurs expérimentés ont vu l'outil comme approprié à une utilisation expérimentale et pour l'apprentissage de la musique. Quatre des sept utilisateurs déclarèrent qu'ils aimeraient avoir une application similaire en leur possession pour composer ou juste expérimenter.

Les utilisateurs ont principalement aimé : l'interface zoomable, l'idée de visualiser les paramètres de génération sous forme d'objet, de pouvoir éditer les paramètres dynamiquement par des courbes en vue zoomée, de pouvoir éditer les clips avec la fonctionnalité de surlignage des notes dans les accords et les gammes définies, de pouvoir re-générer les clips, de pouvoir visualiser l'historique des générations et de pouvoir se repérer dans le réseau de nœuds. La possibilité de pouvoir sauvegarder, puis plus tard importer, un ensemble d'objets $clip$ et d'objets *paramètres de génération* sélectionnée a aussi été jugé pertinente.

Dans de nombreux cas, la rencontre au complet durait une heure, un peu plus qu'initialement prévu. Mais plusieurs utilisateurs se sont montrés volontaires pour utiliser le prototype plus

longuement ou pour avoir la possibilité de le réutiliser une nouvelle fois à une prochaine rencontre. Certains encore ont demandé s'il était possible d'avoir une copie du prototype. Deux utilisateurs ont passé un total de deux heures chacun à l'utiliser. Ces séances plus longues ont duré grâce à la curiosité dont les utilisateurs ont fait preuve pour comprendre le fonctionnement du système.

Certains utilisateurs se sont montrés réellement émerveillés par celui-ci comme en témoigne le discours suivant de l'un d'entre eux après la rencontre : «Je vais sans doute en rêver cette nuit. Il me tarde dimanche pour la tester à nouveau, en ayant un peu digéré l'apprentissage du fonctionnement de l'interface.» Il a dit avoir été un peu abasourdi, surpris agréablement, par le concept. Mais aussi déstabilisé. Il savait que ça servait à générer des mélodies, mais n'y croyait pas vraiment. Il a été surpris de voir à quoi ça pouvait ressemblait et de constater que les résultats étaient satisfaisants. Il est à noter que cet utilisateur n'avait aucune connaissance de l'état de l'art de la recherche dans la génération de musique.

4.1.3 La discussion

Les tests utilisateurs révèlent que notre système semble adapté aux utilisateurs intermédiaires. Bien que méritant certainement des améliorations, il leur convient dans son ensemble et ils jugent qu'il serait utile. Les utilisateurs débutants ont été convaincus de l'intérêt d'un tel système et se sont montrés très curieux d'en apprendre le fonctionnement. Malheureusement ils se sont trouvé en difficulté dans une étape du processus du choix des paramètres de génération : le choix des gammes et de la progression d'accord. Le système d'écriture de la progression d'accords à partir d'une cadence et d'une gamme choisies leur a permis de passer l'étape, mais un peu péniblement, leur donnant l'impression de ne pas maitriser ces paramètres définissant l'harmonie. Un système assistant l'utilisateur dans le choix des accords et des gammes apparait comme nécessaire pour rendre l'usage de GenSession satisfaisant pour les utilisateurs débutants. Certains suggérèrent des solutions, par exemple une interface qui proposerait des progressions d'accords ou juste des accords, avec des aperçus sonores ou un système de vi-

sualisation visuel, et la possibilité de glisser-déposer les accords proposés dans la progression d'accords, puis de les réorganiser, les éditer.

Les utilisateurs ayant une connaissance poussée du solfège ont trouvé le système intéressant, ludique, efficace pour répondre à la problématique, mais jugent, pour eux, le recours aux algorithmes de composition automatique inutiles pour composer. Ils pensent cependant que des applications dans le domaine de l'éducation seraient à explorer. Ces utilisateurs expérimentés cependant que l'utilisation de GenSession est intéressante pour l'expérimentation et pensent qu'une telle application offre un axe de réflexion intéressant sur le processus de composition.

Tous les utilisateurs ont relevé comme points forts du prototype : le re-encadrage automatique de l'espace de travail, le système de visualisation de l'historique, la fonctionnalité de re-génération active et la possibilité de changer l'harmonie d'un clip déjà généré avec la *transposition intelligente* laissant entrevoir des possibilités d'agir sur les paramètres harmoniques d'un $clip$ après génération.

À la question « Est-ce que selon vous cette interface est une bonne solution pour l'utilisation d'algorithmes de génération de mélodies ? »Tous ont répondu oui et certains ont même rajouté que c'était une bonne solution pour l'édition de séquences MIDI en général. Du moins le concept de la vue globale (scène) permet de naviguer dans un ensemble de séquences qu'il est possible de dupliquer, de supprimer, de déplacer, de fusionner et de glisser-déposer sur la timeline pour construire une séquence plus longue. Les utilisateurs ont trouvé que l'interface zoomable permettait de naviguer intuitivement entre les vues, en plus de pouvoir visualiser toute ou une partie des objets.

CONCLUSION

Des travaux tels que ceux de Povel (2010) et Sneyers et De Schreye (2010) nous ont montré que des algorithmes de génération de mélodies, basés sur les règles, peuvent donner des résultats suffisants pour satisfaire des utilisateurs. Mais les interfaces utilisateurs permettant de choisir les paramètres de génération et d'évaluer les résultats des générations, par de nombreux aspects, limitent leur utilisation à l'expérimentation de la capacité de ces générateurs.

C'est à partir de ce constat que nous nous sommes demandé comment une interface utilisateur pourrait rendre accessible l'utilisation de générateurs de mélodies à des utilisateurs débutants tout en satisfaisant des utilisateurs intermédiaires, permettre la visualisation et l'édition des paramètres de génération et des résultats des générations, permettre la visualisation de l'historique des paramètres utilisés pour générer les mélodies, permettre la réutilisation de mélodies précédemment écrites ou générées dans de nouvelles générations.

Nous avons développé GenSession, un outil de musique générative à l'interface utilisateur novatrice, afin de permettre à des musiciens sans connaissance préalable en génération de musique ou en théorie musicale de générer des mélodies, de les visualiser, de les modifier, de les recombiner, de les rassembler sur différentes voix d'une même séquence, de les mettre les unes à la suite des autres pour construire une séquence plus longue, et de les réutiliser dans de nouvelles générations. Cette interface devait permettre à l'utilisateur de visualiser et d'éditer les séquences musicales générées et les paramètres de génération utilisés pour les générés, et de pouvoir visualiser l'historique des générations permettant à l'utilisateur de déterminer quels paramètres, et éventuellement quelle mélodie préexistante, ont été utilisés pour générer une mélodie.

La représentation des paramètres de génération et des passages de musique générés sous forme d'objets placés dans un espace de travail 2D (appelé *scène*) permet différentes interactions sur les séquences musicales et les paramètres ayant servi à les générés, et servent de base à plusieurs fonctionnalités de visualisation. L'historique des générations, copies et créations de $clip$ est visualisé grâce aux liens entre les clips. Nous pensons que ces liens entre les $clips$

aident aussi l'utilisateur à se repérer dans un réseau important de $clips$. Cela permet aussi de visualiser tout ou parti des objects présents sur l'espace de travail grâce à un système de re-encadrage automatique de la *scène*, mais aussi grâce à un système de zoom et de panoramique manuel. Ensuite, cela permet de basculer vers une vue zoomée des objects permettant une édition avancée de leur contenu tel que la configuration des paramètres dynamique par le tracé de courbes pour les objets *paramètres de génération* ou l'ajout/suppression des notes et la transposition vers une nouvelle harmonie pour les objets *clips*.

GenSession tire aussi avantage des concepts des interfaces utilisateur zoomables (Zoomable User Interfaces ou ZUI, Bederson et Hollan (1994)), permettant à l'utilisateur de zoomer sur un clip (pour visualiser le contenu dans le détail et l'éditer) ou dézoomer pour avoir une vision globale de la scène et voir tous les clips, les objets paramètres de génération et les liens qu'il y a entre eux. L'utilisateur peut basculer entre ces différentes vues en pressant simplement un raccourci grâce au système de zoom automatique. Le passage entre les vues se fait par une transition animée, permettant à l'utilisateur de se repérer entre les deux niveaux d'abstractions. Lorsque l'utilisateur est dézoomé et qu'il bouge des clips sur l'espace de travail 2D, la scène ajuste le zoom et la position afin de maintenir la visibilité de tous les clips en tout temps.

GenSession autorise une variété importante de paramètres que l'utilisateur peut choisir et qui peuvent ensuite être utilisés pour paramétrer un algorithme de génération. Les paramètres peuvent être des booléens, des entiers, des nombres à virgule flottante, et même des paramètres qui peuvent varier dans le temps que nous appelons *paramètres dynamiques*. De tels paramètres dynamiques peuvent être dessinés, sous forme de courbe, avec la souris et par l'utilisateur, autorisant ce dernier à expérimenter comment réagit un algorithme de génération à la fluctuation des valeurs des paramètres dans le temps.

Notre interface permet aussi de générer un $clip$ en gardant une partie des informations contenues dans un $clip$ existant comme le rythme ou la hauteur des notes pour ensuite re générer les informations qui manquent à partir de n'importe quel object *paramètre de génération*. Le nouveau $clip$ est alors lié à deux objects parents : le $clip$ dont il hérite, par exemple, le rythme ainsi que l'object *paramètres de génération* dont il hérite les paramètres harmoniques (progression

d'accords, gammes, etc.). On pourrait voir ces techniques comme se rapprochant de la notion de mutations propres aux algorithmes génétiques (Unehara et Onisawa (2003)) où l'on réutilise la matière précédemment écrite ou générée dans de nouvelles itérations du processus de génération. D'ailleurs le processus de ces algorithmes génétique peut être reproduit avec GenSession si l'utilisateur le souhaite. Cela contribue a montrer que notre interface est $flexible$ car elle à l'utilisateur de varier ses processus de création dans l'utilisation du prototype : ici on n'oblige pas l'utilisateur à suivre un processus entièrement prédéfinit comme avec d'autres prototypes basés sur les algorithmes génétiques tel que celui de Unehara et Onisawa (2003).

Au-delà des fonctionnalités liées à la manipulation des paramètres de génération et des résultats des générations, GenSession offre une nouvelle façon de manipuler et d'éditer des passages de musique MIDI, d'une part avec le système de zoom pour passer d'une vue de la scène à la vue d'édition d'un $clip$ mais aussi par certaine fonctionnalité implémentée qui sont des spécificités du système comme le système de $transposition\ intelligente$ qui permet de transposé une séquence de notes d'une progression d'accords et de gammes vers une nouvelle harmonie en gardant la fonction de ces notes dans la nouvelle harmonie. Cette fonctionnalité permet donc aussi d'agir rapidement sur les paramètres de génération définissant l'harmonie sur une séquence précédemment générée sans avoir à la re générer. Ce système de transposition d'une harmonie vers une nouvelle harmonie, avant d'être une fonctionnalité d'édition d'un $clip$, a été développé à la base pour assurer soutenir l'algorithme $variante\ 2$ gardant la hauteur des notes d'un $clip$ et changeant la durée des notes, leur position temporelle et donc déplaçant possiblement certaines notes de mesure.

Les retours d'utilisateurs, ayant des niveaux divers de connaissances en musique, nous montrent que GenSession a plu aux testeurs. Les possibilités offertes par l'interface zoomable laissent la place à un large horizon de possibilités, certaines inattendues, et celles-ci se placent comme une réponse adéquate à l'utilisation de générateurs de mélodies. Le paradigme zoomable est très apprécié. La navigation dans le réseau de nœuds représentant les liens entre les entités $clips$ et *paramètres de génération* est rendue agréable et l'édition du contenu des objects représentés par les nœuds semble facilitée par le système de zoom et panoramique automatique. Les

tests utilisateurs ont montré que GenSession, dans sa version actuelle, est adapté au besoin des utilisateurs intermédiaires désireux d'interagir avec des générateurs de mélodies. Ceux-ci se sont montrés à l'aise avec l'ensemble des fonctionnalités. Les utilisateurs débutants ont eu des difficultés dans la phase de choix des accords et des gammes, incontournable pour utiliser notre générateur de mélodie basé sur les règles. Si l'on souhaite rendre l'interface adaptée aux utilisateurs débutants, il faut impérativement implémenter un système qui rende possible le choix des accords et des gammes sans connaissance du solfège. Peut-être qu'il faudrait même un système de suggestion automatique les progressions d'accords. Les utilisateurs confirmés ont jugé que l'utilisation de générateurs de mélodies ne leur était pas destinée. Mais ces derniers ont reconnu les qualités du prototype.

Les perceptives de recherches futures et les fonctionnalités que nous souhaiterions voir implémenter dans le futur sont énoncées dans la section suivante.

RECOMMANDATIONS

Les perspectives de recherches futures

Pour l'avenir, nous avons imaginé plusieurs axes de développement autour du prototype qui seraient pertinents afin de développer les concepts de notre interface un peu plus loin.

Afin de tester la flexibilité de notre interface, en vue de l'améliorer et de découvrir l'étendue des possibilités qu'elle propose, nous aimerions implémenter d'autres méthodes de génération et observer comment leur utilisation en est simplifiée ou complexifiée. Certaines de ces méthodes pourraient être basées sur des techniques différentes que l'utilisation d'algorithmes basés sur les règles. Pourquoi ne pas essayer de réutiliser les données contenues dans un ensemble de clips précédemment écrits, générés ou importés au cours d'une session ? Nous pourrions par exemple imaginer une méthode de génération rendue sélectionnable lorsqu'un certain nombre de $clips$ seraient sélectionnés sur la scène (peut-être une dizaine de $clips$ apporteraient suffisamment d'informations ?). Cette méthode exécuterait un algorithme exploitant une mémoire musicale qui serait la somme des séquences de l'ensemble des clips sélectionnés. Cet algorithme pourrait être basé sur les chaines de Markov (Schulze et van der Merwe (2011)). Il pourrait aussi bien utiliser les algorithmes de mutation du matériel génétique développés autour des algorithmes génétiques appliqués à la génération de musique comme Unehara et Onisawa (2003) et Biles (1994). On pourrait même utiliser directement des algorithmes génétiques avec une évaluation algorithmique à chaque itération basée sur un modèle de Markov comme Lo et Lucas (2006) utilisant les n-grammes. Nous pourrions aussi implémenter un algorithme tels que celui de Hamanaka *et al.* (2008) qui permet la fusion de deux mélodies monophoniques. Nous pourrions ainsi générer un $clip$ qui serait un mélange de deux autres $clips$ parents en pouvant choisir le pourcentage à garder de chacune. Aussi, il serait pertinent d'implémenter d'autres algorithmes de génération basés sur les règles tels que ceux de Gwee (2002). Comme Papadopoulos et Wiggins (1999) l'a dit, originellement en anglais, « cela deviendra de plus en plus commun de combiner différentes méthodes et tirer avantage des forces de chacune. »

Un deuxième axe de recherche serait d'essayer de rendre notre prototype plus adapté à l'utilisateur débutant en améliorant le système d'harmonisation pour aider l'utilisateur à choisir les progressions d'accords. La progression d'accords est un paramètre déterminant et lors des évaluations, c'est le paramètre qui s'est avéré le plus compliqué à configurer pour les utilisateurs débutants et intermédiaires. Il y a plusieurs systèmes qui pourraient s'avérer efficients. Une solution serait de générer des progressions d'accords aléatoirement ou peut-être les suggestions pourraient être générées par un processus partiellement aléatoire et possiblement basé sur des paramètres (décrivant la tension désirée par exemple, telle que le «jazz factor» et le «happy factor» de Simon *et al.* (2008)). Une autre solution serait un système qui laisse l'utilisateur choisir des accords avec un ensemble d'outils qui permettent à un utilisateur avec peu ou pas de connaissance en solfège de faire un choix qui le satisfait. Ces outils pourraient être une fonctionnalité de pré-écoute des accords et des gammes (une version amélioré de la fonctionnalité apportée par le bouton «preview» qui ouvre une boite de dialogue permettant d'écouter les accords de la progression d'accords, voir section 3.5.1) ou aussi bien un système de suggestions de gammes et de cadences. Dans la version actuelle du prototype, l'utilisateur peut choisir un accord pour chaque mesure. Ce serait bien d'aller plus loin en lui permettant de choisir des accords différents pour chaque temps de chaque mesure comme c'est souvent le cas dans la musique occidentale.

Un troisième axe de recherche serait de travailler sur la gestion d'un nombre de clips sur l'espace de travail qui peut devenir relativement important et nuire à la visibilité. Plus il y a de $clips$ sur la scène, plus ceux-ci apparaissent petit et plus les liens qu'il y a entre eux ont tendance à se croiser. Afin d'aider les utilisateurs à gérer une large collection de clips, des caractéristiques pourraient être ajoutées pour réorganiser la disposition des clips automatiquement basés sur l'algorithme de graphe de Di Battista *et al.* (1999). Les utilisateurs pourraient aussi, après les avoir sélectionnés, réduire un groupe de clips (en anglais «collapse») en «méta nœud». De plus, des techniques venant des interfaces de bureau virtuel qui permettent à l'utilisateur de rassembler des icônes dans des piles (Mander *et al.* (1992), Agarawala et Balakrishnan (2006)) pourraient être adaptés pour travailler avec des ensembles de clips. Ceci nous amène à soulever des problèmes relatifs à l'idée de mettre les «miniatures» représentant l'information musicale

du contenu des clips dans une pile : comment donner un aperçu du contenu d'une pile de clips ? Nous pourrions imaginer que lorsque l'utilisateur passe la souris au-dessus d'une pile de clips, un clip est choisi au hasard ou un sous-ensemble est sélectionné par un processus algorithmique «intelligent», puis est joué.

Un quatrième axe de recherche pourrait être de chercher un moyen de mieux visualiser l'historique des créations, générations, copies et fusions d'objets $clip$ et d'objets *paramètres de génération*. L'utilisateur pourrait obtenir des informations complémentaires sur la nature des liens entre les objets. Un chemin à explorer commencerait par l'implémentation d'une solution pour visualiser des informations supplémentaires sur la façon dont un clip a été généré comme la méthode ou algorithme de génération utilisé. Cette information est par exemple gardée en mémoire par le programme, il suffirait de trouver le meilleur moyen de l'afficher. Peut-être pourrions nous la faire apparaitre dans une petite boite lorsque la souris survole le lien entre deux objets, tels que le lien entre un objet *paramètres de génération* et un objet $clip$.

Un cinquième axe de recherche serait de trouver un système d'interaction et de visualisation permettant de conserver ou de re-générer seulement une partie de la séquence musicale d'un $clip$ existant. Une solution serait, en vue zoomée sur un $clip$, de pouvoir sélectionner un ensemble de notes sur le pianoroll et faire ensuite apparaitre par un clic droit un menu radial qui proposerait de copier les notes sélectionnées ou de les re-générer. Mais dans ce cas, comment conserver l'historique ?

Un sixième axe de recherche serait de renforcer la cohérence des liens de parenté en tout temps, mais en permettant toujours de modifier les paramètres d'un objet parent lié à un ou plusieurs enfants. Je m'explique. Si vous générez un objet $clip$ à partir d'un objet *paramètres de génération* et que vous souhaitez ensuite modifier les paramètres de l'objet *paramètres de génération*, actuellement vous pouvez. C'est déconseillé puisque vous perdriez ainsi l'historique en rompant symboliquement le lien entre les paramètres parents et son $clip$ enfant précédemment généré. Pour conserver toutes les informations liées à l'historique des générations, la solution actuelle est de copier l'objet *paramètres de génération*, puis de modifier sa copie. Ce processus résout le problème en conservant une bonne logique. Mais peut-être pourrions-nous trouver

une meilleure solution qui éviterait de copier l'objet parent, ce qui allégerait le processus et éviterait un encombrement de l'espace de travail si trop d'objets venaient à être copiés pour ces raisons. Si l'on décide d'appuyer ce processus, dans ce cas, il faudrait interdire à l'utilisateur de modifier un objet qui a été utilisé pour une génération et ainsi rendre obligatoire sa copie avant modification. Sinon une autre approche, si l'on accepte qu'un objet parent soit modifié, serait de changer l'aspect des liens pour préciser que des changements ont été apportés. On pourrait, par exemple, mettre les liens en pointillé si le lien de parenté a été fragilisé par une modification du parent (ou même de l'enfant).

Finalement, des fonctionnalités pour aider la compréhension par l'utilisateur des différences entre deux clips pourrait être bénéfique, peut-être en surlignant les différentes notes d'une paire de clips sélectionnés, pareillement à la façon dont différentes versions d'un fichier texte sont visualisés avec un outil «diff». Surligner les différences dans les séquences de notes de clips peut aider l'utilisateur à voir immédiatement où deux clips diffèrent sans avoir à les écouter. En plus de cela, surligner les différences dans les progressions d'accords ou de tonalité entre deux clips pourrait aider l'utilisateur à vérifier si deux clips sont «compatibles» pour être fusionnés dans un nouveau clip. Le surlignage des différences pourrait être résolu par un «rollover» : le clip sous le curseur et tous ses clips voisins pourraient avoir leurs différences surlignées. Ces différences pourraient aussi être visualisées au niveau d'un ensemble de clips : l'utilisateur bénéficierait de la visualisation montrant comment une scène a évolué au travers du temps, montrant quels clips ont été créés et supprimés. Des techniques du précédent travail Archambault *et al.* (2011) et Zaman *et al.* (2011) sur la visualisation des graphes dynamiques pourraient être une source d'inspiration.

Les fonctionnalités à ajouter

En dehors des axes de recherche cités plus haut, nous avons imaginé de nombreuses fonctionnalités qui pourraient permettre d'améliorer l'expérience utilisateur et d'enrichir GenSession en nouvelles possibilités de création. Voici celles qui nous paraissent les plus pertinentes et originales.

Il serait intéressant de réfléchir à des moyens de générer des paramètres de générations sous forme d'objets paramètres de génération. Ensuite, nous pourrions générer de nouveaux clips à partir de ces nouveaux paramètres. Par exemple, nous pourrions déterminer les paramètres de génération du nouvel objet paramètres de génération à partir d'un clip sélectionné. Ceci demanderait un algorithme qui analyserait la séquence musicale et en déterminerait les paramètres de génération susceptibles de donner un résultat similaire. Nous pourrions aussi déterminer ces paramètres à partir de deux clips différents, en extrayant les paramètres harmoniques de l'une des séquences musicales et les paramètres rythmiques de l'autre. Mais aussi, ajouter une fonctionnalité permettant de générer l'intégralité ou une partie des paramètres de génération de manière aléatoire pourrait donner des résultats intéressants.

Une autre fonctionnalité à implémenter permettrait de créer un $clip$ à partir d'un fichier MIDI importé. Une fois la séquence du $clip$ harmonisée (les accords et gammes choisis pour chaque mesure ou temps de la séquence), les possibilités sont abondantes. Nous pourrions imaginer créer un objet paramètres de génération à partir d'un fichier MIDI ou bien encore d'utiliser le $clip$ importé pour générer un nouveau $clip$ par une méthode de génération qui conserve des informations de celui-ci (telle que la variante 1 de note algorithme, voir 3.10.4). Pour retrouver les gammes et accords d'une séquence importée, nous pourrions imaginer d'implémenter un système d'harmonisation automatique similairement à celui de Simon *et al.* (2008).

Un troisième axe de développement serait d'améliorer la gestion des voix (voir section 3.3) par l'ajout de fonctionnalités telles que la possibilité d'en ajouter ou d'en supprimer via l'interface utilisateur lors de l'exécution du programme et non plus en dur dans le code, avant la compilation, comme c'est le cas actuellement. Les voix pourraient avoir plus de propriétés. Une voix représente en quelque sorte une partie d'une partition qui est destinée à être jouée par un instrument (réel ou virtuel). Ainsi chaque voix pourrait être définie comme $polyphonique$ (à l'image d'une partition de piano, on peut jouer des accords) où $monophonique$ (à l'image d'une partition de flute qui ne peut produire qu'une note à la fois). On pourrait également définir le registre ou la tessiture (note la plus haute et la plus basse pouvant être écrite/joué). Ces paramètres pourraient être pris en compte par les algorithmes de génération, des para-

mètres indépendants des objets paramètres de génération et propres à chacune des voix. La fonctionnalité de transformer une voix polyphonique en voix monophonique, dans la séquence d'un clip existant et selon certains paramètres, serait à ce moment-là utile. La possibilité, en vue d'édition d'un clip, d'afficher ou cacher chacune des voix serait aussi une fonctionnalité bienvenue.

Un quatrième axe de développement serait de chercher à améliorer la timeline. Par exemple en commençant par la fixé sur l'axe des ordonnées et de mettre un slider pour la déplacer sur l'axe des abscisses. Il faudrait aussi améliorer le système permettant de visualiser les liens entre les clips présents sur la timeline et leur représentation sur la scène.

Nous pourrions aussi permettre à l'utilisateur de connecter directement l'interface utilisateur de GenSession avec d'autres modules de génération. Par exemple, nous pourrions interfacer GenSession avec SoundHelix de Schürger (2013) en générant un fichier XML résumant les paramètres de génération pour SoundHelix et les passages musicaux générés au bon format pour que SoundHelix puisse générer un morceau de musique au complet. Ou encore nous pourrions développer un système de plug-in qui puisse permettre de développer indépendamment des algorithmes de génération selon une certaine norme, puis de les utiliser dans le prototype.

Pour finir, l'utilisation de la fonctionnalité de re génération active, qui a été très appréciée des utilisateurs, soulève le problème suivant : il n'y a aucun moyen, en vue dézoomée sur la scène ou sur la timeline, de différencier un $clip$ qui se re-générera à chaque lecture, d'un clip statique. Il faudrait trouver une solution pour permettre de visualiser ce changement d'état du clip. Car il s'agit bien d'un changement d'état puisque le $clip$, qui lorsqu'il est statique représente une séquence de notes et des options de configuration, représente alors une référence aux paramètres, méthodes de génération et éventuellement $clip$ parents qui permettront de le re-générer. C'est un peu comme si le nœud enfant n'était plus vraiment un $clip$ mais une projection dynamique de ces parents au travers de la méthode de génération.

ANNEXE I

SCÉNARIO DE PRÉSENTATION DE L'APPLICATION

Scénarios standard de présentation de GenSession.

- Création d'un objet paramètres de génération :
  - Création de l'objet.
  - Zoom sur l'objet
  - Configuration de chaque paramètre un par un, lentement.
    Accords : E minor I V II I.
  - Dessin de la courbe de densité et du pourcentage des notes dans l'accord (vs note dans la gamme).
  - Dé-zoom
  - Génération d'un clip
  - Lecture du clip en cliquant sur «play».
  - Zoom on clip
- Création d'un objet clip :
  - Changement des accords, puis retour en arrière.
  - Changement des accords.
  - Lecture du clip avec nouveaux accords.
  - Retour en arrière avec accords d'origine.
  - Démonstration des différentes options du panel.
  - Activation de l'option de re génération active.
  - Lecture du clip (avec re génération).
- Génération d'accords :
  - Dé-zooom

  - Création d'un nouvel object paramètres de génération.
  - Zoomé dessus.
  - Choisir le nom de l'objet : «chords»
  - Configurer les paramètres pour générer des accords.
  - Dé-zoom
  - Sélectionner la voix rouge.
  - Générer un clip et le jouer.
  - Selectionner «keep blue voice», selectionner sur la scène une mélodie sur une voix bleue, générer un clip et le jouer.

- Fusion de clips : Prendre un clip avec une voix bleu, prendre un clip avec une voix rouge (et de préférence avec des accords), et les fusionner.
- Timeline :
  - Glisser-déposer des clips avec différents nombres de mesures, sur la timeline. (Maybe 3, not too many)
  - Jouer la timeline
  - Supprimer, insérer et replacer clips sur la timeline.
- Sauvegarder la session :
  - Ouvrir une session : montrer que tous les objets et les liens entre les objets sont récupérés et charger sur la scène.
  - Importer selement quelques objets sur la scène en plus.
  - Sélectionner un ensemble d'objets et les sauvegarder.
- Rediriger le signal MIDI vers un autre MIDI device
  - Ouvrir le menu de configuration du MIDI et sélectionner le port virtuel vers lequel rediriger les messages MIDI durant la lecture. Puis lancer une application tierce écoutant le port midi virtuel précédemment sélectionné et configurer pour faire jouer les notes par des instruments virtuels.

- Zoomer sur un clip avec deux et les jouer. Cette fois-ci les notes sont reçu par l'application tierce précédemment configurée et chaque voix peut être jouée par un instrument virtuel différent.

ANNEXE II

NOTES PRISES AU COURS DES RENCONTRES

Certaines de ces notes ont étaient tronquées de phrases jugées inutiles, mais la quasi-totalité est conservée. Les notes concernant les bugs du prototype observés ont également été supprimées. Des fautes d'orthographe ont été corrigées. Pour préserver leur anonymat, le nom des participants a été remplacé par un numéro d'identification.

1. Rencontre avec le participant ♯1

Ne connaît pas le solfège

Retour positif. Il souhaite revenir ce dimanche pour tester à nouveau l'application, mieux la prendre en main, la tester avec ses applications. Il a été un peu abasourdi, surpris agréablement, par le concept. Mais aussi déstabilisé. Il savait que ça servait à générer des mélodies, mais n'y croyait pas vraiment. Il a été surpris de voir à quoi ça pouvait ressembler et de constater que les résultats étaient satisfaisants. Il a dit «je vais sans doute en rêver cette nuit. Il me tarde dimanche pour la tester à nouveau, en ayant un peu digéré l'apprentissage du fonctionnement de l'interface.» Il a cependant trouvé la prise en main difficile, car beaucoup de concepts à comprendre. Il a également peu de connaissance en solfège, ce qui a rendu l'apprentissage plus long. Il a cependant l'habitude d'utiliser des logiciels qui manipulent du MIDI et n'a pas eu de mal à comprendre comment router les messages vers d'autres logiciels. C'est l'exemple typique d'un artiste qui a une connaissance de la musique timbrage (il a enseigné l'histoire de la musique électronique et de l'électro-acoustique, en école de musique), mais très peu de connaissances du solfège et de la création d'une structure harmonique. Il semblait, bien que trouvant les concepts compliqués, aimer l'interface et avoir une grande motivation pour surmonter les difficultés. Ça a été notamment dur de retenir les raccourcis. J'ai fini par les écrire sur une feuille.

J'ai relevé durant le test les informations suivantes : Question : Un clip fusionné doit-il, pour chacune de ces voix, être dépendant de ses parents (si l'on change de parent, ça change aussi

le fils. Je pense que oui... — Note à moi-même : Fonctionnalité vraiment cool > Pouvoir créer un settings object à partir d'un clip.

2. Rencontre avec le participant ♯2

Musicien amateur

Il finit un master et travaille sur une interface logicielle dans le domaine du contrôle de paramètres pour le son.

Ses remarques : —SPACE est utilisé pour lancer la lecture dans la plupart des séquenceurs logiciels. —Multiselection est un peu bizarre comme notion en dualité avec la notion de sélection simple. —Nécessité d'un raccourci pour arrêter la lecture (barre espace ?) —Paramètres qu'il serait cool de pouvoir ajouter : velocity, volume, syncopation. —Il a beaucoup aimé l'interface et aimerait vraiment beaucoup que le concept d'auto frame soit intégré à Max MSP. —Bonne expérience utilisateur. —Un bon départ pour le développement d'un outil d'aide à la composition plus abouti.

3. Rencontre avec le participant ♯3

Musicien

Le participant a essayé le prototype durant 1 h (dont 20min où je lui est montré le fonctionnement.

Ses remarques :

— the shortcut SPACE for transport is better — Autoframe is cool — He wanted to get a tension variation by playing with parameters. — Miss an help system to get more informations about commands, etc.. — Miss the shorcut BACKSPACE to delete objects on scene. — Number value feedback on the drawer parameter curve. — Give the possibility to write directly the range in the settings object panel options field. Conclusion : Good interface, easy to use, but require a training time like every music production tools to understand all the functioning. I

ask : Do you think it is a good way to integrate generator melody in the composition process ? He answer : Yes, for sure it is. A very good way.

4. Rencontre avec le participant ♯4

Musicologue

His impression : No metrome -> it cool be nice (to evaluated the rhythmic placement of a simple melody) SPACE for zooming is not logical > usually it's used for the transport. Re Generation is cool He like globally the concept. Like to draw parameters and the zoomable interface was easy to use for him.

He told me about HYPERSCORE, an similar application that allow to compose with a scene and object system for non-musician. You can also draw in object but not for parameters but for arrangement. There is no zoom system. He told me about also about a thesis Music, Mind and Health where people try to help sick person with the music generation system, HYPERSCORE.

5. Rencontre avec le participant ♯5

Musicien amateur

Ses commentaires : —voudrait utiliser BACKSPACE pour supprimer un clip. moi > (la densité est plus la durée moyenne au final. Il serait bien sinon que l'algorithme prenne en input une liste des durées autorisées (par ex 3 cases à cocher : notes pointées, ternaires, binaires) + la densité moyenne et sa variance qui représenterais le nombre de notes dans une mesure par exemple). —Améliorer la timeline avec sliders et empêcher les clips d'aller en dessous. —Pouvoir sélectionner quelques notes et pouvoir faire apparaitre un menu radial qui propose de copier les notes ou de les re générer (comme avec la re génération activée).

6. Rencontre avec le participant ♯6

Musicien / guitariste amateur

Ses commentaires : —aimerait bien un feedback visuel sur le choix de la densité. —Aimerait pouvoir diminuer ou augmenter d'une octave, à la fois dans la tessiture dans les settings objects, mais aussi pour les clips déjà générés. —Bug système de sauvegarde > impossible de fusionner après avoir chargé une ancienne session. —Aimerait pouvoir voir la cadence qui correspond aux accords choisis. —Bug copy object settings > le «num of notes in the same time unit» est mal copié.

Son avis : He like globally> He like getting help to compose. Un bon outil pour générer des idées sans se passer de l'instrument pour lui qui est musicien amateur. Il a l'habitude de composer en commençant par choisir des accords alors il trouve que le concept est adapté a sa façon de travailler. —Il a peur qu'il y ait trop d'éléments sur la scène lors d'une grosse session. Il aimerait un système de sliders qui permette de bouger sur la scène après avoir zoomé manuellement dessus (au lieu d'utiliser SHIFT + clic pour se déplacer. —L'aperçu miniature des clips est très cool. —Un système de «grid» transparent qui permet de «quantifier» l'emplacement des clips sur la scène serait un plus pour pouvoir réorganiser les clips rapidement. Cette personne aime que ce soit bien rangé, d'autre non, il faut que se soit une option. —Il aimerait avoir plus de temps pour explorer plus. L'entretien a duré 1 h 20 et il a beaucoup aimé. Il voudrait avoir une application comme ça et serait même prêt a aider au développement d'un tel logiciel open source. —Il a beaucoup aimé la possibilité de router le signal MIDI et la possibilité d'enregistrer la scène, mais aussi le fait de pouvoir travailler sur plusieurs voix.

7. Rencontre avec le participant ♯7

Ne connait pas le solfège ou la musique

SPACE plutôt utilisé pour le transport BACKSPACE pour supprimer un clip serait bien Difficile de choisir une cadence ou une progression d'accord pour le participant qui n'a aucune connaissance en musique (le participant fait un PHD en art) Difficulté a retenir les raccourcis. Densité rythmique pas logique, car les notes les plus longues devraient être vers le bas Elle a découvert une fonctionnalité assez cool : si on génère plusieurs voix avec plusieurs notes sur la même unité de temps pour chacune on arrive a obtenir une mélodie polyphonique très riche.

Difficulté a choisir la tessiture juste avec le numéro d'octave. Elle aurait aimé un meilleur repère visuel. Dans l'ensemble elle pense que c'est facile a utiliser, au vu des résultats, mais trouve simplement qu'il est dur de choisir des accords qui répondent aux attentes avec cette interface sans connaissances en musique.

8. Résumé des rencontres avec les testeurs

—Aimerait bien un feedback visuel sur le choix de la densité. —Un système de «grid» transparent qui permet de «quantifier» l'emplacement des clips sur la scène serait un plus pour pouvoir réorganiser les clips rapidement. Certaines personnes aiment que ce soit bien rangé, d'autres non, il faut que se soit une option. —Pouvoir choisir des accords différents pour chaque beat en cliquant directement sur l'objet settings sur scène en gardant la possibilité de générer une progression d'accord à partir d'une cadence.

Remarques : —commencer par faire le choix des accords semble être une démarche assez naturelle pour les testeurs ayant des connaissances musicales. Les autres ont plus de mal a comprendre le pourquoi. —Un système plus évolué pour le choix de la cadence serait le bien venu. Il faut assister un maximum l'utilisateur dans le choix des accords, sans le brider, car c'est la seule étape qui demande vraiment, dans l'état actuel du prototype, des connaissances en solfège. —Les gens adorent le concept de ZUI et le fait d'avoir l'aperçu du contenu des clip en miniature qui permet a tout le monde de bien se repérer. —Aimerait avoir une telle application chez lui. +4/7 —Les raccourcis sont importants > ils n'ont pas été très bien choisis, il faudrait par exemple éviter d'utiliser BACKSPACE pour le zoom in on clip. —Le système de zoom rend l'interface simple à comprendre et à utiliser. —Il faudrait absolument rendre la timeline visuellement séparée de la scène. Par exemple mettre un spider entre la timeline et la scène qui puisse permettre de bouger la timeline de gauche à droite et il faudrait quelle soit fixe sur l'axe des y. —Plus que le zoom, l'autoframe est très apprécié. —Une personne m'a dit qu'elle rêverait de voir l'autoframe sur les environnements de programmation visuelle telle que MaxMSP. —Un bon départ pour le développement d'un outil d'aide à la composition plus abouti. —Multiselection est un peu bizarre comme notion en dualité avec la notion de sélection simple. —Paramètres qu'il serait cool de pouvoir ajouter : velocity, volume, syncopation. -

Good interface, easy to use, but require a training time like every music production tools to understand all the functioning.

BIBLIOGRAPHIE

Ableton. 2013. *Ableton Live*, version 9. Logiciel. Germany : Ableton.

Agarawala, Anand et Ravin Balakrishnan. 2006. « Keepin' It Real : Pushing the Desktop Metaphor with Physics, Piles and the Pen ». In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI)*. p. 1283–1292.

Anders, Torsten, Michael Alcorn, et Christina Anagnostopoulou. 2003. *Composing music by composing rules : Computer aided composition employing constraint logic programming*. Technical report. Technical report, Sonic Arts Research Centre, Queen's University Belfast.

Archambault, Daniel, Helen C. Purchase, et Bruno Pinaud. 2011. « Animation, Small Multiples, and the Effect of Mental Map Preservation in Dynamic Graphs ». *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, vol. 17, n° 4, p. 539–552.

Ariza, Christopher. 2005. « An Open Design for Computer-Aided Algorithmic Music Composition : athenaCL ». PhD thesis.

Arpège Musique. 2013. « Arpège Musique - Logiciel pour imprimer et composer ses propres partitions ». En ligne. <http://www.arpegemusique.com/arpege.htm>. Consulté le 4 novembre 2013.

Assayag, Gérard, Camilo Rueda, Mikael Laurson, Carlos Agon, et Olivier Delerue. 1999. « Computer-assisted composition at IRCAM : From PatchWork to OpenMusic ». *Computer Music Journal*, vol. 23, n° 3, p. 59–72.

Bederson, Benjamin B. et James D. Hollan. 1994. « Pad++ : A Zooming Graphical Interface for Exploring Alternate Interface Physics ». In *Proc. ACM Symposium on User Interface Software and Technology (UIST)*. p. 17–26.

Biles, John A. 1994. « GenJam : A genetic algorithm for generating jazz solos ». In *Proceedings of the International Computer Music Conference (ICMC)*. p. 131–131.

Cabrol, François. 2013. « GenSession : A Flexible Zoomable User Interface for Melody Generation ». En ligne. <http://hifiv.ca/~francoiscabrol/GenSession/>. Consulté le 29 août 2013.

Cabrol, François, Michael J. McGuffin, Marlon Schumacher, et Marcelo M. Wanderley. 2013. « GenSession : a Flexible Zoomable User Interface for Melody Generation ». In *Proceedings of the 10th International Symposium on Computer Music Multidisciplinary Research*. p. 542–553.

Di Battista, Giuseppe, Peter Eades, Roberto Tamassia, et Ioannis G. Tollis, 1999. *Graph Drawing : Algorithms for the Visualization of Graphs*. Prentice-Hall.

Digidesign. 1991-2013. *Pro Tools*, version 11. Logiciel. Tewksbury (Massachusetts) :Avid Technology.

Furnas, George W et Benjamin B Bederson. 1995. « Space-scale diagrams : Understanding multiscale interfaces ». In *Proceedings of the SIGCHI conference on Human factors in computing systems*. p. 234–241. ACM Press/Addison-Wesley Publishing Co.

Gannon, Peter. 1990. *Band-in-a-Box*, version 2.0. Logiciel. Victoria (BC) : GPMusic.

Gwee, Nigel. 2002. « Complexity and heuristics in rule-based algorithmic music composition ». PhD thesis, Department of Computer Science, Louisiana State University, Baton Rouge, Louisiana.

Hamanaka, Masatoshi, Keiji Hirata, et Satoshi Tojo. 2008. « Melody morphing method based on GTTM ». In *Proc. of ICMC*. p. 155–158.

Han, Jinseung. 2011. « Digitally Processed Music Creation - Music composition approach utilizing music technology ». PhD thesis, Teachers College, Columbia University, New York, New York.

Image-Line. 2013. « Piano Roll Randomize Tool ». En ligne. <http://www.image-line.com/support/FLHelp/html/pianoroll_random.htm>. Consulté le 30 août 2013.

Intermorphic. 2013. *Noaktil*, version 2. Logiciel. Sunnyvale (California) : Intermorphic.

Keup, Jessica F. 2011. « Computer Music Composition using Crowdsourcing and Genetic Algorithms ». PhD thesis, Graduate School of Computer and Information Sciences, Nova Southeastern University, Broward County, Florida.

Lo, Man Yat et Simon M Lucas. 2006. « Evolving musical sequences with N-Gram based trainable fitness functions ». In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*. p. 601–608. IEEE.

MacCallum, Robert M, Matthias Mauch, Austin Burt, et Armand M Leroi. 2012. « Evolution of music by public choice ». *Proceedings of the National Academy of Sciences*, vol. 109, n° 30, p. 12081–12086.

MakeMusic. 1988-2012. *Finale*, version 2012. Logiciel. Eden prairie (MN) : MakeMusic.

Mander, Richard, Gitta Salomon, et Yin Yin Wong. 1992. « A 'Pile' Metaphor for Supporting Casual Organization of Information ». In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI)*. p. 627–634.

Morwaread, Farbood, Kaufman Henry, et Jennings Kevin. 2007. « Composing with hyperscore : An intuitive interface for visualizing musical structure ». In *International Computer Music Conference*.

Papadopoulos, George et Geraint Wiggins. 1999. « AI methods for algorithmic composition : A survey, a critical view and future prospects ». In *AISB Symposium on Musical Creativity*. p. 110–117.

Povel, D.J. 2010. « Melody Generator : A Device for Algorithmic Music Construction ». *Journal of Software Engineering & Applications*, vol. 3, p. 683–695.

Raskin Software. 2012. *Raskin | Beyond Desktop*, version 1.8. Logiciel. Zurich (Switzerland) : Raskin Software.

Raynaud, Philippe. 2000-2013. « Audiofanzine : Home Studio, MAO, Guitares, Basses, Sono et DJ - Audiofanzine ». En ligne. <http://fr.audiofanzine.com/>. Consulté le 3 novembre 2013.

Ruelle, David, 2008. *L'Étrange beauté des mathématiques*. Odile Jacob.

Schulze, Walter et Brink van der Merwe. 2011. « Music generation with Markov models ». *Multimedia, IEEE*, vol. 18, n° 3, p. 78–85.

Schürger, Thomas. 2013. « Introduction | SoundHelix ». En ligne. <http://www.soundhelix.com/>. Consulté le 4 novembre 2013.

Simon, Ian, Dan Morris, et Sumit Basu. 2008. « MySong : automatic accompaniment generation for vocal melodies ». In *Proc. ACM Conference on Human Factors in Computing Systems (CHI)*. p. 725–734.

Situngkir, Hokky. 2010. *Exploitation of memetics for melodic sequences generation*. Technical report. Bandung, Indonesia : Bandung Fe Institute.

Sneyers, Jon et Danny De Schreye. 2010. « APOPCALEAPS : Automatic music generation with CHRiSM ». In *11th International Society for Music Information Retrieval Conference (ISMIR 2010), Utrecht, The Netherlands (August 2010) Submitted*.

Sneyers, Jon, Wannes Meert, Joost Vennekens, Yoshitaka Kameya, et Taisuke Sato. 2010. « CHR (PRISM)-based probabilistic logic learning. ». *TPLP*, vol. 10, n° 4-6, p. 433–447.

Steinberg. 1990-2012. *Cubase*, version 7. Logiciel. Hammerbrook (Germany) : Steinberg Media Technologies.

Sweers, Marieke, Makiko Sadakata, et Louis G Vuurpijl. September 2012. « New rules for the Melody Generator II Repeating rhythm and varying harmony ».

Unehara, Muneyuki et Takehisa Onisawa. 2003. « Construction of music composition system with interactive genetic algorithm ». In *Journal of the Asian Design International Conference*.

Van Wijk, Jarke J et Wim AA Nuij. 2003. « Smooth and efficient zooming and panning ». In *IEEE Symposium on Information Visualization (InfoViz), 2003.* p. 15–23. IEEE.

Vandenneucker, Dominique. 2011. *Pizzicato*, version 3.6. Logiciel. Verviers (Belgium) : Arpège Musique.

Wikifonia. 2013. « lead sheets ». En ligne. <http://www.wikifonia.org/>. Consulté le 2 septembre 2013.

Zaman, Loutfouz, Ashish Kalra, et Wolfgang Stuerzlinger. 2011. « The Effect of Animation, Dual View, Difference Layers, and Relative Re-Layout in Hierarchical Diagram Differencing ». In *Proceedings of Graphics Interface (GI)*. p. 183–190.