



HAL
open science

Représenter les traces numériques des pièces d'informatique musicale pour faire de l'analyse.

Maxence Larrieu

► **To cite this version:**

Maxence Larrieu. Représenter les traces numériques des pièces d'informatique musicale pour faire de l'analyse.. 2014. hal-01883449

HAL Id: hal-01883449

<https://hal.science/hal-01883449>

Preprint submitted on 27 Oct 2018

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

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



Distributed under a Creative Commons Attribution 4.0 International License

REPRÉSENTER LES TRACES NUMÉRIQUES DES PIÈCES D'INFORMATIQUE MUSICALE POUR FAIRE DE L'ANALYSE.

Maxence Larrieu
Université Paris-Est
LISAA, EA 4120
maxence@larri.eu
orcid.org/0000-0002-1834-3007

RÉSUMÉ

Nous nous intéressons aux œuvres composées dans le domaine numérique. À l'aide du concept de tripartition nous expliquons la présence de traces résultantes du processus compositionnel. Après avoir amené des outils conceptuels pour s'appropriier ces traces, nous proposerons un outil informatique capable de les représenter. Un tel support permet une saisie de l'œuvre indépendamment de sa réalisation sonore. Nous souhaitons ainsi augmenter les moyens d'analyse de ces œuvres.

1. INTRODUCTION

L'usage des technologies du vingtième siècle dans la composition musicale a engendré de nombreux bouleversements. Avec ces technologies la réalisation sonore d'une œuvre n'implique plus nécessairement une partition traditionnelle. Dès lors se pose le problème de l'analyse : comment analyser une œuvre uniquement à partir de sa réalisation sonore ?

Le concept de tripartition sera utilisé pour en lumière les difficultés de l'analyse de ces œuvres. À la suite d'une première approche théorique sur un des niveaux de la tripartition, nous développerons des outils permettant à l'analyste de saisir l'œuvre, indépendamment de sa réalisation sonore. Enfin, nous intégrerons ces avancées dans un outil informatique permettant de construire une représentation des traces numériques.

2. DES TRACES POUR L'ANALYSE

2.1. Tripartition sémiologique

En envisageant le musical comme fait symbolique, Jean Molino et Jean Jacques Nattiez ont développés dans les années 1970 la sémiologie de la musique. Leur démarche se fonde sur une tripartition du fait musical :

Le phénomène musical (...) ne peut être correctement défini ou décrit sans que l'on tienne compte de son triple mode d'existence,

comme objet arbitrairement isolé, comme objet produit et comme objet perçu.¹

Cette tripartition, véritable fer de lance de leur démarche, donne lieu à « *trois familles d'analyses* »² du phénomène musical : à la production de l'objet correspond *l'analyse poïétique*, à sa perception l'analyse *esthétique* et à l'objet « arbitrairement isolé » correspond *l'analyse du niveau neutre*.

Si l'étude de la musique comme fait symbolique est propre à une démarche sémiologique, nous constatons que le concept de tripartition s'avère fécond pour l'analyse musicale. Notamment il permet à Jean Jacques Nattiez de rassembler et de comparer différentes analyses musicales effectués et surtout d'en dégager différents fondements.

Nous envisageons cette tripartition comme une richesse pour l'analyste car elle lui permet de délimiter différents niveaux qui sont autant de moyen de saisir l'œuvre à des fins analytiques. Aussi, pouvons nous envisager, que la mise en relation des différentes vues analytiques obtenues, par les divergences et convergences résultantes, puisse permettre une appréhension plus profonde de l'œuvre³.

Corollairement, en se questionnant sur le mode d'être de l'œuvre musicale, Jean Jacques Nattiez est amené à souligner la limite d'une « approche unidimensionnelle »

L'être de l'œuvre est donc réparti entre quatre instances, couvertes par les pôles de la tripartition. Cette ontologie de l'œuvre sera capitale pour notre propos analytique, car elle

¹ [8], p.37

² [11], p.37

³ sans pour autant prétendre à une ultime compréhension de l'œuvre. Jean Molino souligne : « l'œuvre est à l'horizon de toutes les réécritures possibles » [11], p. 98

montre clairement que toute analyse d'une œuvre particulière reste nécessairement en deçà de son être propre, mais surtout, elle indique qu'on ne saurait, pour en rendre compte, se contenter d'une approche unidimensionnelle.⁴

2.2. Des traces

La délimitation du poïétique et de l'esthétique est aisément compréhensible du fait notamment de la différence de fonction entre le compositeur et l'auditeur. Cependant, pour l'instance intermédiaire, nommé *niveau neutre*, la délimitation est davantage difficile. Nattiez le définit ainsi :

L'œuvre dans sa réalité matérielle (production sonore, partition, texte imprimé, etc.) c'est à dire la trace physique résultant du processus poïétique⁵.

La difficulté réside dans la référence à la matérialité de l'œuvre car, dans la musique écrite, celle-ci est plurielle : relevons au moins la partition et les actualisations sonores. En conséquent, le niveau neutre n'étant pas délimitable *a priori* dans le processus musical, c'est toutes les analyses qui découlent des niveaux qui pourraient être « bancales ». L'auteur, bien conscient de cette ambiguïté, différenciera deux cas selon l'intégration ou non de la partition dans le niveau neutre.⁶

Pour nous permettre de recourir à la tripartition en dehors d'une démarche sémiologique, et en considération de l'ambiguïté du niveau neutre, nous emploierons le terme de *trace* pour référer à ce niveau intermédiaire. Nous envisageons les traces doublement, d'abord comme les *résultantes du poïétique* et ensuite comme les *éléments nécessaires à l'actualisation de l'œuvre*.

En sus, la différence que nous souhaitons amener entre les traces et le niveau neutre, se trouve dans le rapport entretenue avec l'œuvre. Dans la sémiologie musicale le niveau neutre est envisagé comme « la matérialité de l'œuvre » et qui permet « l'analyse des configurations immanentes de l'œuvre »⁷. À la différence, nous envisageons les traces comme les moyens d'actualisation de l'œuvre, indépendamment de quelconque immanence. Si un lien existe entre les traces et l'immanence, actuellement, nous ne saurions le définir.

⁴ *ibid*, p.98

⁵ [11], p.37

⁶ La sémiologie dont il sera à l'origine concevra la partition comme niveau neutre. *ibid*, p100

⁷ C'est nous qui soulignons. *ibid*, p. 37

2.3. Nature des traces

L'usage des technologies du vingtième siècle est venu changer radicalement la nature des traces des œuvres musicales. Nous verrons succinctement la nature des traces pour les œuvres utilisant les technologies électroacoustique, puis numérique.

2.3.1. Traces électroacoustiques

Dans les musiques électroacoustiques ces traces ne se trouvent plus être nécessairement une partition mais un support audio. Stéphane Roy dans un ouvrage portant sur l'application de la sémiologie aux musiques électroacoustiques constate :

Le seul niveau neutre de la musique électroacoustique est donc de nature sonore et non visuelle, et il est contenu dans le support d'enregistrement.⁸

Ces traces de nature sonore, sont dorénavant non écrite, non lisible et coexiste au temps de l'œuvre. Dès lors la délimitation d'un niveau neutre, et *a fortiori* de son analyse - par le fait que nous avons accès à ce niveau que par l'écoute - se trouve fortement augmenté en ambiguïté.

2.3.2. Traces numériques

Plusieurs recherches se basent sur l'étude des traces réalisées dans le domaine numérique. Un enjeu récent est de considérer ces traces pour faire face à l'obsolescence des œuvres ([1], [13]) ; un second enjeu corollaire est d'étudier les traces afin de réaliser des portages – ou reconstruction – des œuvres ([7]). Des analyses musicales peuvent être amenées à la suite d'une reconstruction ([4], [2]) mais nous n'avons relevé aucune recherche systématique sur ces traces à des fins d'analyse musicale.

Afin d'élucider cette possible lacune intéressons-nous à la nature de ces traces résultantes du poïétique.

3. RICHESSE DU NUMÉRIQUE

Pour s'appropriier le numérique le compositeur a recours à différents outils : logiciels interfacés, logiciels de programmation graphique ou textuel de plus ou moins haut niveau, ou encore fichiers audio pré enregistrés. Malgré cette diversité tous ces moyens d'appropriation ont nécessairement un point commun : comme le processeur informatique travaille uniquement avec des nombres binaires, l'ensemble des instructions faites par le compositeur doit être traduit en langage binaire pour être effectif. Dès lors un premier moyen de former des traces numériques serait de récolter l'ensemble de ce langage binaire résultant du poïétique.

⁸ [14] p.28

Un second moyen serait d'employer le même outil utilisé par le compositeur pour accéder aux résultats de son travail. Ce second choix sera fortement privilégié puisqu'il facilite grandement la compréhension des traces⁹.

À la suite de cette approche théorique, élucidons le contenu de ces traces, qui seront aussi désignées par le terme code informatique.

3.1. Instrumentarium

L'usage du numérique que nous ciblons implique inévitablement une production numérique du son nécessaire à la réalisation sonore de l'œuvre. Dès lors en envisageant le code selon sa finalité, nous pouvons le concevoir comme un producteur de signal audio numérique. En excluant le cas extrême où le numérique est utilisé uniquement comme support du signal, nous pouvons avancer que le code informatique a pour fonction de calculer les échantillons constituant le signal à diffuser. Il s'agirait d'un ensemble d'opérations formelles calculant, à la fréquence d'échantillonnage, les échantillons du signal.

Par correspondance fonctionnelle avec l'instrument traditionnel nous appellerons ce contenu du code l'instrumentarium.

Aussi pouvons-nous constater qu'à l'inverse du domaine instrumental, les traces numériques contiennent la construction formelle de l'instrumentarium.

3.2. Variables

Parallèlement à une vision du code selon sa finalité, nous pouvons l'envisager dans sa relation avec le compositeur. Pour composer avec l'instrumentarium le compositeur a besoin de définir un certain nombre de variable qui lui seront nécessaires pour articuler le matériau sonore de sa pièce. Cette notion de variable n'est pas propre au numérique. Nous pouvons par exemple comprendre la partition traditionnelle comme l'affectation dans le temps d'au moins trois variables fondamentales au système occidentale : la hauteur, la durée et l'intensité¹⁰.

Dans un contexte plus large, le principe de variable est aussi utilisé par Molino, et repris par Nattiez, dans le but d'éclairer la production et l'analyse musicale. À la suite d'une citation de Molino, Nattiez souligne :

Les processus poétiques musicaux se fondent sur l'autonomisation et la particularisation de certaines variables.¹¹

Les variables ici amenées englobent les précédentes en ce qu'elles ne se trouvent plus nécessairement en lien avec le phénomène sonore. Elles peuvent en effet être issues de variations « entre le compositeur et l'exécutant, entre le chef et l'exécutant, (...), [ou des] variations sur les gestes, le silence (...) »¹². Ces variables sont aussi variées que ne l'est le *fait musical total* mis en évidence par Molino.

En ne considérant que les variables liées à la production sonore, et par analogie avec le domaine instrumental, nous pouvons noter les différences majeures qu'amène l'utilisation du numérique : les variables (i) ne préexistent pas à la composition : elles sont *fondamentalement indéterminées* (ii) ne sont pas destinées à être interprétées par l'homme (iii) n'ont aucune contraintes physique.

3.3. Complexité

Malgré des similarités de fonctions avec le domaine instrumental, ces traces numériques restent difficilement appréhendables. La construction formelle de l'instrumentarium, ajoutée à la gestion des variables, peut former une quantité d'opérations colossales. À titre d'exemple, la pièce *Partita I* du compositeur Philippe Manoury requiert plus de 40 000 expressions¹³. En supplément de cette densité, l'indétermination des variables implique une quasi infinité de possibilités pour articuler le matériau sonore. Ces spécificités font atteindre aux traces numériques un certain niveau de complexité de tout autre nature qu'auparavant.

4. VERS UNE APPROPRIATION

La présence et la richesse des traces numériques nous laissent entrevoir la possibilité d'une analyse d'un nouvel ordre. Pour permettre une telle analyse une appropriation des traces numériques est nécessaire. En partant maintenant du code informatique – que nous supposons textuel – nous proposerons deux outils conceptuels permettant d'éclairer cette complexité.

4.1. Abstraction

Après de multiples lectures du code informatique, l'analyste repérera différents ensembles de différentes fonctions. Pour envisager ces ensembles en tant que tel il pourra procéder à des abstractions : il s'agit

⁹ Le langage binaire est étranger à l'homme, pour s'approprier le numérique le compositeur utilise un langage de plus haut niveau plus proche de son propre langage (nombre en base dix, langage alphanumérique, syntaxe...)

¹⁰ Sans les énumérer, Nattiez écrit : « Tout l'histoire des notations musicales n'est d'ailleurs pas autre chose que le passage au niveau graphique des variables qui sont indispensables pour garantir "l'éternité" de la musique. » [11], p. 106

¹¹ [11], p. 180

¹² [8], p. 180

¹³ Frédéric Dufeu dans un article portant sur la visualisation de patch Max mentionne la présence de 40 111 objets Max contenu dans le patch d'exécution de *Partita I* [5]

d'identifier les éléments constitutifs d'un même ensemble, de repérer les variants et invariants présents dans cet ensemble, et enfin de considérer l'ensemble en lui-même. Ce procédé n'est pas propre à notre démarche, dans le contexte informatique il est envisagé par Yann Orlarey comme une des six dimensions cognitives des langages de programmation¹⁴. Ainsi il est fortement probable que le code informatique développé soit déjà structuré avec des abstractions.

Un exemple peut être ici amené dans le langage de programmation Pure Data qui propose nativement l'usage d'abstractions. Nous tenons cependant à différencier le procédé d'abstraction théorique que nous avons amené et son implémentation dans ce langage spécifique.

L'ensemble d'opérations présenté en figure 1 permet de générer et contrôler une enveloppe ADSR. Par le procédé d'abstraction, l'utilisateur pourra créer une nouvelle boîte permettant de regrouper cet ensemble. Cette nouvelle boîte (cf figure 2) comporte un nom (`adsr~`) et les arguments permettant de paramétrer l'enveloppe (respectivement peak level, attack time, decay, sustain et release). Par ce procédé d'abstraction, l'ensemble initial de six boîtes est encapsulé en une nouvelle boîte.

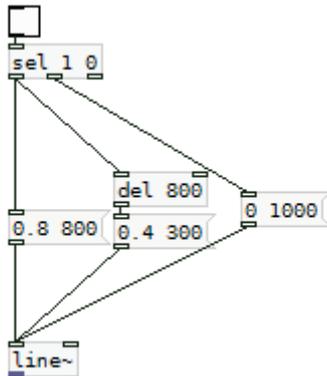


Figure 1. Patch permettant de générer une enveloppe ADSR



Figure 2. Abstraction permettant de générer une enveloppe ADSR

4.2. Différencier

Un second procédé peut être amené en vue d'augmenter la lisibilité du code. Une distinction formelle entre deux éléments autorisera une appréhension plus aigüe. Parmi les nombreuses lignes de code nécessaires à la formalisation de l'instrumentarium, nous pouvons différencier deux éléments : ceux qui *calculent* les échantillons audio numériques à diffuser, et ceux qui *paramètrent* ces calculs. Cette distinction peut être appuyée par le rapport entretenu avec les échantillons : les éléments de calcul sont en rapport *direct* avec ces derniers (la valeur des échantillons résulte du calcul) tandis que les éléments qui paramètrent les calculs sont en rapport *indirect* avec les échantillons.

Un exemple concret peut être amené dans le langage de programmation Csound. Soit la ligne de code suivant permettant la génération d'une sinusoïde :

```
ares oscils iamp, icps, iphs
```

Les éléments `iamp`, `icps` et `iphs` – nécessaires pour déterminer respectivement l'amplitude, la fréquence et la phase à l'origine – sont ceux qui paramètrent le calcul. À l'inverse, `oscils` est la fonction de calcul des échantillons. A

Nous appellerons *moteur audio* le regroupement des éléments de calculs, et *contrôle* celui des paramètres nécessaires aux calculs.

Des distinctions similaires sont présentes dans les langages de programmation audio : dans Csound les variables de signal audio (`a name`) sont à différencier de celles de contrôle (`k name`) et possèdent leur propre fréquence de renouvellement¹⁵ ; dans Pure Data une distinction majeure est faite entre les signaux audio et les signaux de données. Nous proposons notre propre terminologie d'abord parce qu'elle diffère légèrement de la classique distinction audio/donné, en ce que nous situons en amont du signal audio (i.e. l'élément qui le calcul), et aussi pour pouvoir appliquer notre distinction à l'ensemble des langages de programmation.

5. VERS UNE REPRÉSENTATION

L'appréhension des traces numériques est potentiellement réalisable par l'analyste. Cependant pour permettre une vue analytique *per se* l'analyste doit pouvoir se référer aux traces de sorte que ses lecteurs puissent comprendre le sens de son propos. Comme il est inconcevable de demander aux analystes de maîtriser l'étendue incessamment croissante des outils informatiques, nous proposerons un nouveau support se substituant à cette diversité. Ce support devra se situer entre le code informatique et la signification analytique,

¹⁴ Parmi la modularité, la consistance, la lisibilité, l'expressivité et la viscosité [12]

¹⁵ <http://www.csound.com/manual/html/OrchKvar.html>, visité en mai 2014

de sorte que l'analyste puisse se référer au code et que ses références puissent être porteuses de sens pour ses lecteurs.

Un parallèle peut être fait ici avec la sémiologie appliquée aux musiques électroacoustiques. L'analyste face à cette trace sonore évanescence (cf I.3.2.) se trouve sans prise pour développer une analyse. Pour faire face à ceci il construira une *transcription* des sons. C'est à l'aide de ce nouveau support qu'il pourra développer une analyse du niveau neutre.

Plus largement Nattiez souligne :

[lorsque la partition est absente] ou ne garantit pas le résultat sonore il faut procéder à une transcription des sons, toujours indispensable, car il faut pouvoir désigner ce dont elle parle (...) ¹⁶

Pour former ce nouveau support nous avons développé un outil informatique qui permet de représenter les traces numériques. La relation entre le code et cette représentation est *a priori* indéfinie. C'est dans la construction du support que l'analyste définit la *distance* nécessaire entre le code et la représentation. Il y a donc autant de représentations possibles que d'appropriations différentes. Nous détaillerons ci-après les choix effectués pour cadrer les représentations.

5.1. Directives globales

5.1.1. Cibler l'interactivité

Nous avons privilégié la représentation des traces d'œuvres se voulant interactives. Nous utilisons ce terme pour spécifier que l'évolution des variables du code informatique est, dans une certaine mesure, déduite de phénomènes extérieurs au code. Ainsi une forme d'interactivité est décelable entre un phénomène extérieur et l'actualisation de la pièce. Intégrer ces éléments extérieurs, que nous considérerons sensibles, dans la réalisation de l'œuvre, demande à l'analyse une dimension supplémentaire. Il faudra tenir compte de ces phénomènes, comprendre leurs provenances et la façon dont ils sont intégrés au code. C'est parce que la représentation des traces de ces œuvres se trouve augmentée en signification que nous avons choisi de les traiter en premier lieu.

5.1.2. Représentations dynamique

Pour que la représentation prenne tout son sens elle doit nous permettre de rendre compte de l'évolution des variables utilisées par le compositeur. Cette particularité demande à la représentation d'intégrer une dimension temporelle. Pour remplir cette condition

nous avons choisi d'afficher la valeur des variables à l'instant *t*. Ainsi la représentation évolue au cours du temps de l'œuvre. Cette caractéristique nous permet aussi une meilleure lisibilité : les éléments non actifs au moment *t* peuvent ne pas être représentés.

A l'inverse d'un support statique qui ne varie pas au cours du temps, nous qualifierons de *dynamique* la représentation visée.

5.2. Choix de réalisations

5.2.1. Représenter l'ensemble

À la suite des distinctions amenées jusqu'ici nous avons choisi de représenter l'ensemble à l'aide de trois éléments : des modules, des paramètres et des connexions. L'ensemble résultant d'une abstraction pourra être intégré dans un module pour lequel l'utilisateur choisira un nom et les paramètres significatifs à afficher. Les dépendances de variations entre différents paramètres seront représentées par des connexions. La distance de la représentation par rapport au code se fait dans le nombre de modules qu'elle inclut.

entre la représentation et le code se fait selon le nombre de module à intégrer.

5.2.2. Subdiviser les modules

Afin d'augmenter la précision de la représentation nous avons différencié trois types de modules : synthèse, traitement et entrée. Les modules d'entrée sont utilisés pour représenter les phénomènes extérieurs qui ont été reliés aux variables du code. Les modules de synthèse et de traitement correspondent à deux opérations possibles pour construire l'instrumentarium. La synthèse calcule les échantillons à partir uniquement de paramètres, tandis que les traitements nécessitent des échantillons préexistants et des paramètres.

5.2.3. Pour une lisibilité

En considération du nombre important de modules à insérer nous avons choisi de ne pas représenter les signaux audio. Ainsi seules les connexions entre paramètres, soulignant une dépendance de variation, seront visibles. Pour signaler qu'un module de traitement agit à la suite d'un signal préexistant nous les placerons simplement en série, l'un en dessous de l'autre.

La représentation d'un module de synthèse ou traitement est faite sur une seule ligne, les paramètres se trouvent à l'intérieur de parenthèses qui succèdent au nom du module. La représentation des modules d'entrée se fera sur deux lignes, la première contiendra le nom du module - elle permettra de référer à la provenance

¹⁶ [11] p.101

du phénomène extérieur - et la seconde contiendra le nom et la valeur du phénomène capté.

Des *points de connexion* situés avant chaque paramètre permettront d'effectuer les connexions.

Une capture d'une représentation arbitraire est présentée en figure 3. Une lecture générale de celle-ci peut être la suivante : les informations d'amplitude et de hauteur sont extraites d'une source nommée « vibraphone stéréo ». Un module de synthèse par modulation de fréquence (FM) suivi d'un traitement de délai génère le flux audio. La fréquence porteuse de la synthèse FM est liée à l'amplitude de la source vibraphone stéréo.

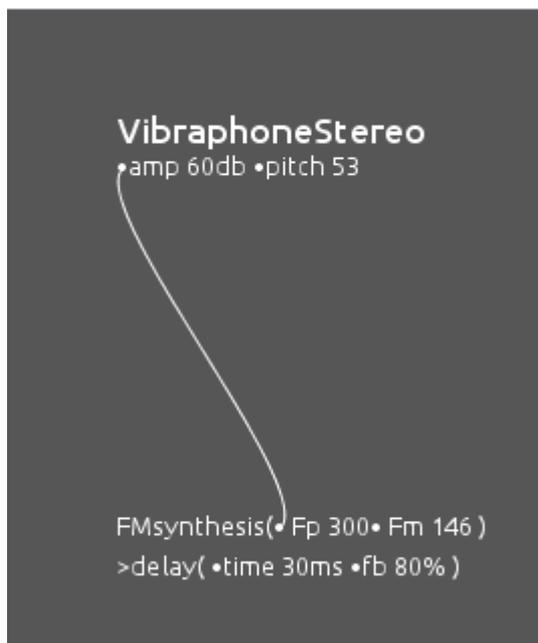


Figure 3. Capture d'une représentation arbitraire réalisée avec l'outil développé.

6. IMPLÉMENTATION

6.1. Un outil ad-hoc

La réalisation de l'outil informatique s'est faite selon une directive principale : être indépendant des langages de programmation audio. Ainsi l'outil pourra être utilisé quel que soit le langage utilisé par le compositeur. L'implémentation s'est faite avec le langage Processing¹⁷, basé sur le langage JAVA, qui est dédié aux arts graphiques. Nous nous assurons ainsi de trouver une solution à nos besoins de représentation graphique.

¹⁷ <http://processing.org/>, visité en mai 2014

6.2. Le standard Open Sound Control

La communication entre l'outil, que nous appellerons ShowProcess¹⁸, et le langage de programmation audio se fait par l'intermédiaire de message Open Sound Control¹⁹ (OSC). Ce protocole de communication nous permet d'une part de s'assurer de l'utilisabilité de l'outil – les messages OSC sont des standards dans la programmation audio – et d'autre part d'être aisément utilisable. L'analyste n'aura pas à apprendre un nouveau langage de programmation mais devra simplement comprendre la structure des messages à utiliser.

6.3. Structure des messages

Un message OSC destiné à un module devra comporter, le type de module ciblé, son nom, ainsi que ses paramètres. Un paramètre se définit à l'aide de deux variables, un pour le nommer et un pour indiquer sa valeur. Trois paramètres dédiés à l'affichage du module ont été créés par défaut : `state`, la variable d'état du module (un module ne s'affiche que s'il est actif), `xpos` et `ypos` les positions cartésiennes du module. La syntaxe des messages à suivre varie selon le type de module .

6.3.1. Modules de type synthèse et entrée

La syntaxe à respecter pour ces modules est la suivante :

```
/moduleType (xpos.ypos) /moduleName /argumentName argumentValue
```

Les champs `moduleType` et `moduleName` précisent le type et le nom du module ; `argumentName` et `argumentValue` le nom et la valeur de l'argument :

6.3.2. Modules de type traitement

La syntaxe des modules de traitement doit inclure le nom du module à la suite duquel il va s'insérer (indiqué par le champ `parentName` ci-après) :

```
/treat/parentName/moduleName/argumentName argumentValue
```

6.3.3. Messages de connexion

La syntaxe des messages destinés aux connexions doit préciser les paramètres à connecter, ainsi que l'état de la connexion.

¹⁸ Un site internet avec un tutoriel et de l'aide à été réalisé. DOI doi.org/10.6084/m9.figshare.7145894.v2

¹⁹ <http://opensoundcontrol.org/>

/action/moduleName(argumentName)/moduleName(argumentName)

Le champ `action` est à remplacer par `connect` ou `disconnect` selon la connexion souhaitée.

6.4. Implémentation

La librairie `oscP5`²⁰, permettant de gérer les messages OSC, a été ajoutée à Processing. La réception des messages se fait par protocole UDP en interne. `ShowProcess` est conçu en utilisant la programmation orientée objet : les modules sont des instances de trois classes `{input,synth,treat}` et les paramètres sont des variables des instances. Ainsi peut être comprise la syntaxe des messages :

```
/className/objectName/objectVariable  
variableValue
```

7. CONCLUSION

En partant du concept de tripartition nous avons ciblé une possibilité pour approfondir l'analyse des œuvres composées avec le numérique. Des outils conceptuels ont d'abord été amenés pour autoriser une appropriation, puis un outil informatique a été développé pour permettre la construction d'une représentation. Ce nouveau support doit permettre une vue analytique spécifique de l'œuvre, fondée non plus sur une réalisation sonore mais sur les traces nécessaires aux réalisations sonores.

Pour finaliser le chemin parcouru, il nous faut dans un futur proche développer une analyse à partir des représentations que nous avons rendues possibles. Cette application nous permettra de nous rendre compte des éventuelles modifications à apporter et surtout de juger du potentiel des représentations.

L'outil développé est à ce stade un prototype, destiné aux musiques avec dispositifs électroniques. Si cette direction, ainsi que les choix de représentations nous sont propres, nous sommes convaincus que la considération des traces numériques permet d'augmenter l'analyse musicale. Des telles analyses peuvent déjà avoir lieu (cf 2.3.2) mais celles-ci sont actuellement dépendantes du portage d'une œuvre. Il nous paraît opportun d'autoriser cette vue analytique, si singulière, indépendamment d'une reconstruction de la pièce. De telles analyses permettraient de développer les connaissances relatives à la composition de ces pièces musicales.

8. RÉFÉRENCES

[1] Barkati, K., Y. Orlarey, J. Barthelemy, « Abstraction du processus temps réel », *Revue*

²⁰ <http://www.sojamo.de/libraries/oscP5/>, visité en mai 2014

Francophone d'Informatique Musicale [En ligne], n° 2, 2012

<http://revues.mshparisnord.org/rfim/index.php?id=184>, visité en mai 2014

[2] BAUDOIN, « la faktura, « outil conceptuel d'analyse » – illustration avec *Stria*, de John Chowning », *Actes des Journées d'informatique musicale*, Grenoble, France, 2009

[3] Dahan, K. *Domaines formels et représentations dans la composition et l'analyse des musiques électroacoustiques*, thèse, université Paris 8, 2005

[4] Dahan, K. « Surface Tension : Dynamics of Stria », *Computer Music Journal*, Vol. 31, n° 3, p. 65-74, 2007. DOI [10.1162/comj.2007.31.3.65](https://doi.org/10.1162/comj.2007.31.3.65)

[5] Dufeu F. « Comment développer des outils généraux pour l'étude des instruments de musique numériques ? », *Revue Francophone d'Informatique Musicale* [En ligne], n° 3, 2013 <http://revues.mshparisnord.org/rfim/index.php?id=255>, visité en mai 2014

[6] Fober, D., Y. Orlarey, S. Letz, « INScore An Environment for the Design of Live Music Scores », *Proceedings of the Linux Audio Conference*, Graz, Autriche, 2013

[7] Lemouton, S., K. Weissenbrunner, A. Bonardi, « A la recherche de Diadèmes, de Marc-André Dalbavie », *Musimédiane* [En ligne], n°6, 2011 <http://www.musimediane.com/spip.php?article133>, visité en mai 2014

[8] Molino, J. « Faits musical et sémiologie de la musique », *Musique en jeu*, n° 17, p. 37-62, 1975

[9] Morin, E. *Introduction à la pensée complexe*. Seuil, Paris, 200

[10] Nattiez, J.J. *Fondements d'une sémiologie de la musique*. Union Générale d'Éditions, Paris, 1975

[11] Nattiez, J.J. *Musicologie générale et sémiologie*. Christian Bourgeois, Paris, 1987

[12] Orlarey, Y. « Entre calcul, programmation et création », *Le calcul de la musique*, Pottier Laurent (dir.), Publications de l'université de Saint-Étienne, Saint-Étienne, p.331-365, 2009

[13] Vincent, A., A. Bonardi, F. Rousseaux, « Modéliser et représenter la création audio-numérique pour la pérenniser », *Revue Francophone d'Informatique Musicale* [En ligne], n° 2, 2012

<http://revues.mshparisnord.org/rfim/index.php?id=230>,
visité en mai 2014

[14] Roy, S. *L'analyse des musiques électroacoustiques : modèles et propositions*. L'Harmattan, Paris, 2003

[15] Wright, M. *The Open Sound Control Specification 1.0* [En ligne], 2002
http://opensoundcontrol.org/spec-1_0, visité en mai 2014
