Transcription de fichiers MusicV vers Csound au travers de OpenMusic

António De Sousa Dias

Transcription de fichiers MusicV vers Csound au travers de OpenMusic

António de Sousa Dias
Université Paris VIII
Centre de Recherche Informatique et Création Musicale (CICM)
2, rue de la Liberté - 93526 - Saint-Denis cedex 02 - France
asousadias@mshparisnord.org

Résumé

Dans ce texte nous présentons un outil de traduction de fichiers MusicV vers Csound programmé dans l'environnement OpenMusic. Bien que cet outil soit limité à la partie «score » il permet le traitement des sous-routines PLF. Nous présentons aussi les raisons et intérêt de la conception de ce genre d'outils pour la conservation d'oeuvres.

Mots clés

Analyse musicale, composition, Csound, modélisation, Music V, Open Music, représentation, synthèse sonore, transcription

Introduction

Parmi d'autres développements, Csound [Vercoe 1986] a connu une considérable amélioration à travers l'introduction des « macros » ainsi que de l'évaluation des expressions numériques au niveau de la partition ([Boothe et al. s.d.] et [Boulanger 2000]). Cependant, la conversion des fichiers MusicV vers Csound n'est pas directe en ce qui concerne le recours aux sous-routines PLF disponibles sur Music V.

Ainsi, le patch <M5_translator_kernel> pour l'environnement OpenMusic [Agon 1997] porte toute son attention à ce niveau là. Pour la vérification de ce travail, nous avons choisi d'accomplir la codification d'extraits des oeuvres ou des exemples de Jean-Claude Risset, les raisons ont été les suivantes : premièrement, ces oeuvres et exemples représentent des exemples très importants dans le champ de la musique par ordinateur, tant du point de vue compositionnel que du point de vue musicologique. Deuxièmement, ces exemples sont documentés et en même temps très répandus parmi la communauté musicale, ce qui se révèle fondamental pour la détection des erreurs.

Finalement, de ces deux raisons découle l'importance et l'intérêt de la préservation de ces fichiers à travers des différents plates-formes et environnements. Le texte suivant décrit d'une façon abrégée les principales raisons et étapes de cette recherche.

La composition comme fondement d'une recherche

Nous poursuivons depuis longtemps une recherche sur les travaux de Jean-Claude Risset en envisageant l'étude des aspects qui découlent de la problématique d'une maîtrise de la synthèse sonore vis-à-vis de la composition musicale, en adaptant ces même travaux à des différents environnements musicaux tels que Csound, OpenMusic ou Max/MSP (cf. [Risset 2002]).

Ainsi, même si nous avons utilisé dans nos oeuvres électroacoustiques des adaptations de ces « recettes » de synthèse depuis le début des années 90, nos travaux les plus récentes tels que « Estranho movimento para um dia como o de hoje » (2000) ou « TêTrês » (2001), parmi d'autres, en doivent beaucoup à cette recherche, y compris des oeuvres instrumentales. Tel est le cas de « Dói-me o luar » (2001), où des procédés issus de l'analyse harmonique d'un son nous ont permis de réaliser des rapports entre des différents motifs.

Par ailleurs l'engagement personnel dans l'enseignement de la musique électroacoustique et surtout celui concernant la synthèse sonore, nous a naturellement amené vers une recherche sur les travaux de Risset.

Music V et préservation de « partitions sonores »

En conséquence de la réutilisation ou d'adaptation des « partitions sonores » [Risset 1995] pour la composition, et puisque on devrait pénétrer à fond dans le processus d'engendrement des matériaux tel que Risset les prévoyait au-délà de la constatation des résultats, on a suivi la problématique de la maintenance des traces originelles de ce même engendrement.

Ainsi, à notre avis, en ce qui concerne la pérennité des oeuvres la seul préservation du résultat final n'est pas du tout suffisante. La réutilisation de ces partitions numériques, tel que Risset l'a souligné [Risset 1995], nous permet déjà de constituer un corpus opératoire, quoiqu'il manque encore une discussion sur les modalités de préservation.

Or, en nous rapportant spécifiquement aux deux environnements de programmation mentionnés, Music V et Csound, nous vérifions que MusicV peut fonctionner en compréhension (c'est à dire, on peut décrire un processus), au travers des sous-routines PLF, tandis que Csound, jusqu'à l'inclusion des « macros », travaillait toujours en extension (il fallait toujours indiquer explicitement tous les événements).

Parmi les transcriptions vers Csound que nous avons réalisées pendant les années 90, à la suite de [Lorrain 1980], celles qui faisaient recours aux feuilles de calcul (« *spreadsheets* ») comme des intermédiaires, dû à l'usage des sous-routines PLF, perdaient quelque part un aspect très important : le procédé de génération du matériau ainsi que le rapport directe entre les deux fichiers extremes (MusicV – Csound).

Nous pensons donc qu'il serait convenable de retenir, dans la transcription de ces fichiers, l'accès à l'originel y compris la description des moyens de développement du matériau musical, c'est à dire, <u>préserver la compréhension des processus d'engendrement</u>.

Le package « m5_translator_kernel »

<u>Les raisons du choix d'OpenMusic (IRCAM)</u>

Il est presque impossible, aujourd'hui, d'accéder au logiciel MusicV, au contraire de Csound lequel est bien repandu. Toutefois, bien qu'on puisse maîtriser les rapports entre les deux languages, la conversion des fichiers

n'est pas évidente. D'ailleurs, nous avons entrepris la création d'un logiciel de transcription en C++, un projet abandonné, car d'une part il réclamait une maîtrise en programation qui n'est pas notre spécialisation, et d'autre part il risquait de devenir fermé par rapport aux musiciens desireux de modifier la structure interne pour l'adapter à leurs propres besoins.

Ainsi, le choix de l'environnement OpenMusic en tant que médiateur répond à un souci de mise en relation directe avec la musique, c'est-à-dire, il s'agit d'un environnement « orienté musique » ; de plus, il contient des librairies permettant la communication vers Csound.

Finalement, à notre avis, le point le plus intéressant se trouve dans la préservation du style de codage à l'entrée, car cette préservation assure la proximité de la pensée de l'auteur rendant plus facile la détection des erreurs.

Description du contenu du package

Le travail a pris la forme d'un package, constitué par des fichiers OpenMusic, et des fichiers au format « texte-only » constituant des exemples soit pour leur utilisation dans Csound, soit dans OpenMusic.

Les exemples terminés par <_om.sco> ou <_om(plfX).sco> constituent des partitions écrites dans un style hybride Lisp-MusicV. Ceci est dû au fait que la fin d'instruction chez MusicV est normalement marquée par point-virgule, tandis que, chez OpenMusic, chaque ligne d'un fichier est plus lisible si elle est mise entre parenthèses (voir Figure 1).

```
(GEN 0 2 1 1 1)
(GEN 0 1 4 0 0 1 4 0.4 7 0.6 15 0.1 30 0.2 250 0.03 480 0 511)
(GEN 0 1 5 0 0 1 6 0.6 11 0.7 30 0.35 250 0.5 340 0.2 480 0 511)
(GEN 0 1 6 0 0 1 10 0.6 16 0.8 35 0.3 250 0.5 340 0.2 480 0 511)
(NOT    0      2    9.7    4000    130.8    130.8    6    4)
(NOT    0.25   2    9.45   4000    185      185      6    4)
(NOT    1.16   4    8.5    3000    116.5    116.5    5    4)
(NOT    1.5    4    6.7    4000    164.8    164.8    5    4)
(NOT    3      6    6.2    800     98       98       3    5)
(NOT    3.5    6    5.5    500     138.6    138.6    3    5)
(NOT    7      4    4      2000    92.5     92.6     6    6)
(NOT    9      2    3      4000    65.4     65.5     7    6)
(NOT    9.5    2    3      600     98       98       3    5)
(NOT    9.51   4    3      500     138.6    138.5    3    5)
(NOT    9.495  3    3      400     233      233      3    5)
(SEC 14)
```

Figure 1. Exemple de codage hybride Lisp-MusicV

Pour lancer la synthèse de ces documents chez Csound ces fichiers doivent être traduits au préalable au travers du patch <m5_translator_kernel> dans OpenMusic.

Ce patch traduit la partie «score » des listages MusicV dans un équivalent Csound. Il reconnaît aussi les sous-routines PLF tels que PLF2, PLF3, PLF4, PLF5, PLF6 et PLF7, utilisés par Jean-Claude Risset dans son « Catalogue » [Risset 1969], (voir par exemple ses œuvres « Inharmonique » (1977) et « Contours » (1982)).

Pour la comprehension de la syntaxe MusicV, nous nous sommes basés sur [Mathews 1969], en remarquant les devélopements et modifications trouvées en examinant les listages presentés par [Lorrain 1980] et [Di Scipio 2000].

Concernant Csound, bien que nous ayons pris toujours comme base de travail [Vercoe 1986], la présente version prend comme référence Csound/Perf versions 1.4/4.0.3 ainsi que le Manuel d'utilisation diffusé sous forme html [Boothe et al s.d.].

Le package contient aussi des textes Csound représentant des fichiers qui nous avons traduit selon les pistes données par [Risset 1969], [Lorrain 1980] et [Di Scipio 2000]. Des renseignements complémentaires se trouvent dans [Dogde et Jerse 1985]. Concernant la description des sous-routines PLF [Mathews 1969], au delà de l'analyse des listages présents dans le Catalogue [Risset 1969], on trouve des renseignements complémentaires chez [Lorrain 1980] et [Di Scipio 2000].

Installation et structure des patches

D'abord, il faut placer dans l'environnement OpenMusic le dossier <music_v_translatorf> qui contient les fichiers suivants (voir Figure 2) :

- <M5_translator_kernel>, le patch principal,
- <ProcessLine>, <case_tables>, des patchs secondaires
- <gen01>, <gen02>, <gen03>, <gen04>, <gen06>, <gen07>, <gen08>, <gen09>, où sont définis les façons dont les fonctions seront traduites,
- <case_pfl>, <plf02>, <plf03>, <plf04>, <plf05>, <plf06> et <plf07>, les sous-routines qui prennent en charge les traitements PLF.

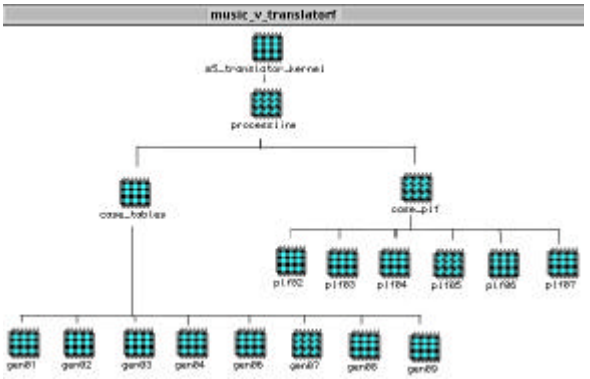

Figure 2. Contenu du dossier « music_v_translatorf » et structure hierarchique.

Un des aspects importants, se trouve dans l'ouverture au changement par l'utilisateur. Si quelqu'un d'autre trouve un moyen plus convenable pour la génération d'une fonction ou pour l'accomplissement d'une tâche, on peut le faire en raison du caractère modulaire ainsi que de l'accessibilité des fichiers. Cet aspect permet des changements locaux sans affectation du codage global.

Au-delà de ces patches, on prévoit un fichier en plus : le fichier de texte <music5_opcodes.txt> qui contient la définition de la conversion entre les codes d'opération et son équivalent numérique (voir Tableau 1). Ceci se révèle fondamental dans le cas où on perd le contenu de la boîte texte.

Finalement, il faut avoir un fichier d'entrée codé dans le style hybride Lisp-MusicV. En bref, dans les lignes suivantes, nous présentons et discutons le comportement de chaque patch.

Description des patches et mode d'opération

Face un usage avec des fichiers déjà testés, les instructions d'utilisation sont assez simples : d'abord, on doit déverrouiller la boîte de texte, importer le texte à transcrire, verrouiller la boîte et, ensuite, évaluer la boîte « editsco ». Ceci declenchera l'ouverture d'une fenêtre de dialogue permetant de sauvegarder le résultat sous la forme d'un fichier texte (voir Figure 3).

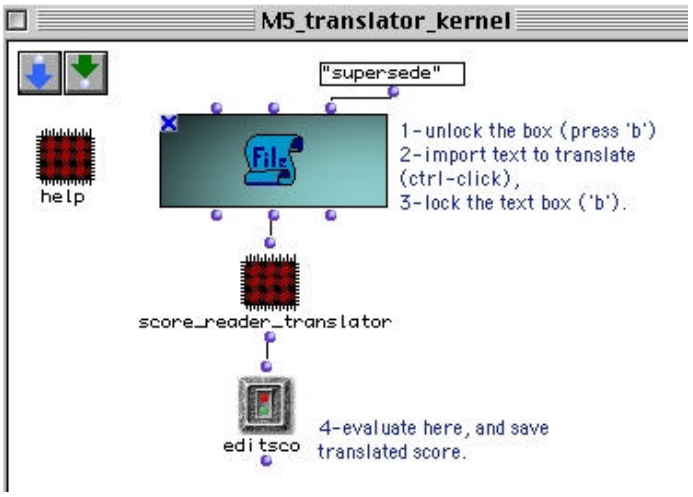

Figure 3. Patch principale

Concernant le fonctionemment interne, le texte est passé vers l'abstraction «score_reader_translator » où le fichier « music5_opcodes.txt » est evalué, le processus comportant les phases suivantes :

1) Traduction des codes alphanumériques en codes numériques selon le tableau suivant ;

(NOT	1)
(INS	2)
(GEN	3)
(SV3	4)
(SEC	5)
(TER	6)
(SV1	7)
(SV2	8)
(PLF	9)
(PLS	10)
(SI3	11)
(SIA	12)
(COM	13)
(MM	70)

Tableau 1. Codage du fichier <music5_opcodes.txt>

2) Répérage des instructions NOT affféctées par des sous-routines PLF ;

3) Traduction de toutes les instructions (une instruction par ligne) dans le format Csound.

Cette dérnière phase est la plus importante, et appelle plusieurs sous-patches externes. Le premier, « ProcessLine », realise le triage des différentes lignes selon l'instruction. Ces lignes seront distribués par des abstractions (case_note, case_ends, case_sv1 et case_MM) et d'autres sous-patches (case_tables et case_plf). L'existence d'abstractions permet d'éviter la prolifération des sous-patches menant à une concentration sur l'essenciel. D'ailleurs, l'accès direct au sous-patch « ProcessLine » permet d'ajouter d'autres instructions sans avoir recours au patch principal.

Quant'aux abstractions, la transcription reste presque directe. Par exemple, dans le cas de l'instruction NOT, elle est substitué par l'instruction «i», et on ne change que l'ordre des champs (MusicV : « NOT » attaque instrument durée ... ; Csound : « i » instrument attaque durée ...) (voir Figure 4).

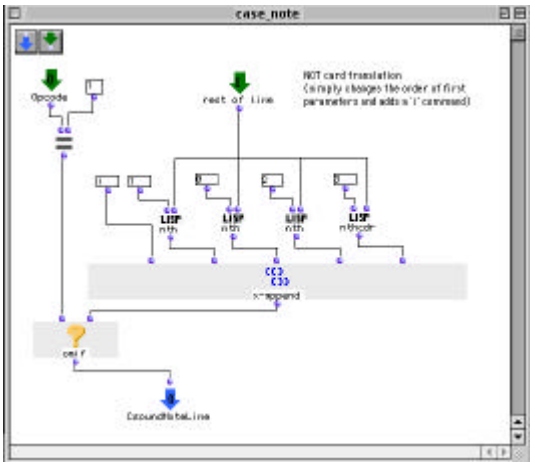

Figure 4. Transcription d'une instruction NOT

La décision de rendre accéssibles les sous-patches dans les cas des tables de fonctions et des sous-routines PLF, obéit à des critères différents. Si, dans le cas des fonctions, il est souhaitable une certaine liberté dû aux différentes possibilités de transcription, dans le cas des sous-routines PLF cette liberté de programmation est même obligatoire car, selon Mathews, les « sous-routines PLF sont des routines de génération et traitement des notes dont le compositeur a l'option de les fournir s'il souhaite se servir de cette possibilité » [Mathews 1969].

Pour les tables de fonctions, les générateurs choisis comprennent les generateurs GEN01, GEN05, GEN07, GEN09 et GEN27. L'utilisation du générateur GEN01 reste limitée à l'appel du fichier « soundin.88 » contenant la fonction en forme de cloche (« *bell shaped function* ») [Risset 1969].

Tous les sous-patches PLF suivent la même logique de construction. Nous esperons ainsi obtenir des conditions qui nous permettent de simplifier la tâche de reprogrammation. Tous les sous-patches ont la même structure d'entrée et de sortie (voir Figure 5). C'est à dire qu'on a comme entrée gauche une ligne de commande « PLF » constituée par les paramètres après p4 et du côté droite la liste des instructions « NOT » affectées. Comme sortie on attend la liste résultante des instructions « i », selon les paramètres d'entrée et de l'algorithme programmé.

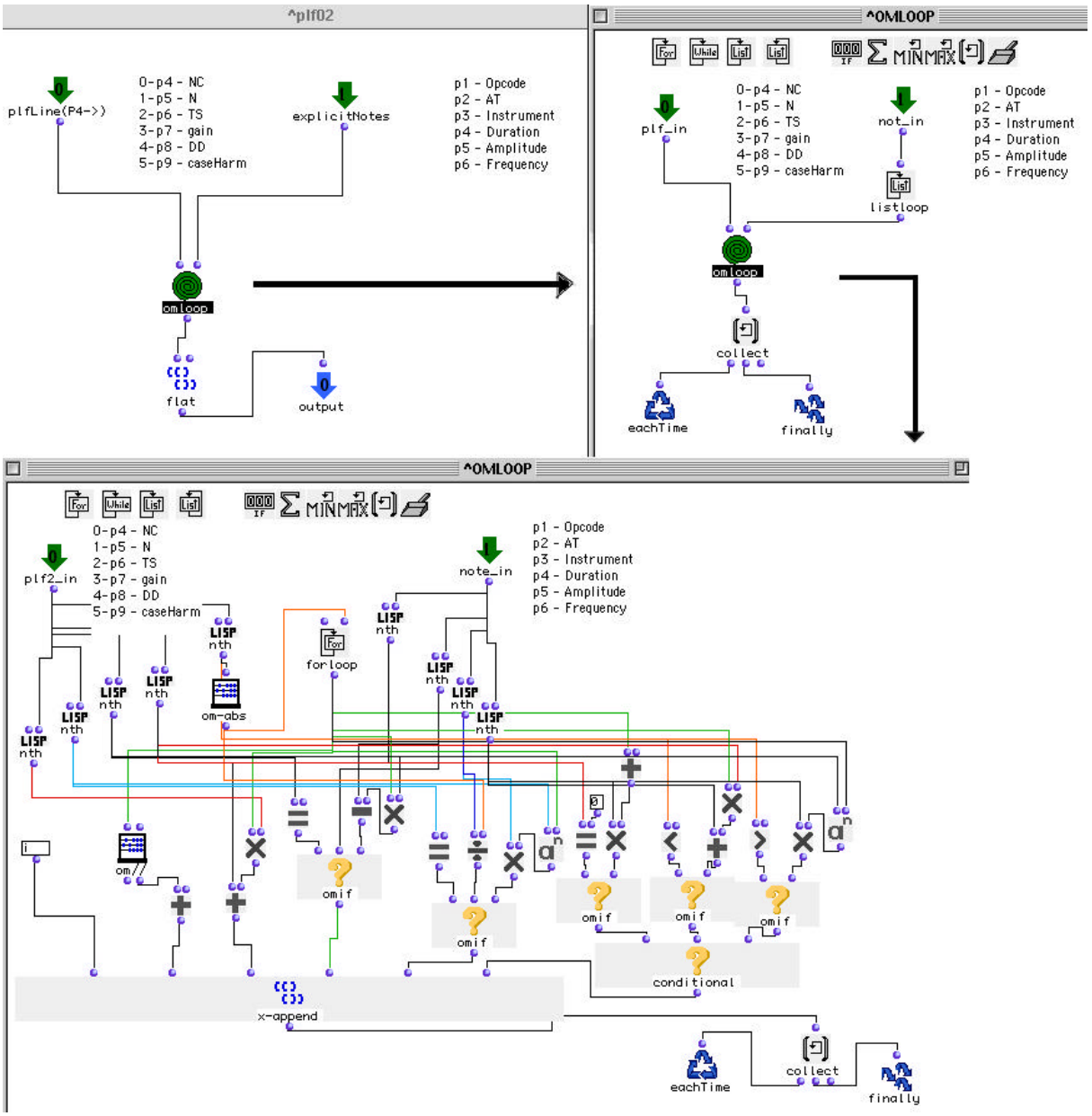

Figure 5. Exemple de sous-patch PLF (Ici PLF02)

Avantages et Limitations

Un des avantages dans l'utilisation d'un système tel que nous l'avons décrit c'est la simplicité du processus de transfert. En effet, il suffit de faire une scanographie et d'effectuer des petits changements dans le fichier MusicV d'entrée. Dans ce cas, on peut remarquer que la détection d'erreurs peut être simplifiée car le fichier d'entrée reste très proche de l'original.

Autre avantage, la principale, concerne l'implémentation des sous-routines PLF. On peut garder la trace de l'idée algorithmique originelle, et en plus on peut visualiser des résultats grâce à l'emploi des boîtes <chord-seq> à des étages intermédiaires, afin de réaliser la traduction la plus convenable.

On regrette néanmoins un désavantage de ce patch dû au fait de ne pas traduire la partie correspondante à l'orchestre Csound. En effet, il faudrait traduire tout le fichier vers un fichier de texte au format « CsoundSynthesizer » en proposant aussi une approche de transcription orchestrale.

Conclusion

Cette approche permet de conclure que la transcription des fichiers MusicV vers Csound, tel que d'autres exemples de transfert technologique, ne relève seulement des intérêts techniques ou technologiques, ni d'une curiosité musicologique, mais plutôt des besoins compositionnels, surtout, en ce qui rélève de la constitution d'un corpus, ce qui constitue une véritable stimulation de la réflexion et expérimentation musicales.

Ainsi, l'étude du travail de Jean-Claude Risset n'est pas pris, ici, par hasard : plus qu'un ensemble de solutions personnelles ou restreintes, il représente une contribution pour les générations postérieures, étant donné le fondement solide dans lequel il s'est développé.

Remerciements

Je dois vivement remercier M. Jean-Claude Risset pour l'intérêt qu'il a révélé dans l'accomplissement de ce travail ainsi qu'à M. Horacio Vaggione, mon directeur de recherche, pour son appui. Des remerciements à Luis Caires pour son aide dans la programmation.

Ce travail a été développé avec le soutien de Fundação para a Ciência e Tecnologia, à Lisbonne.

Références

[Agon 1997] Agon, C. and Assayag, G. (1997). *OpenMusic (OM) 4.4 User's Manual Reference & Tutorial*. Paris : IRCAM - Centre Georges Pompidou.

[Boothe et al s.d.] Boothe, D. ; Boulanger, R. ; ffitch, J. ; Piché, J. (eds.) (s.d.). *The Public Csound Reference Manual — Canonical version 4.01 by Barry Vercoe, Media Lab MIT and contributors*. Version incluse dans le CD-Rom accompagnant [Boulanger 2000].

[Boulanger 2000] Boulanger, R. (ed) (2000). *The Csound Book — Perspectives in Software Synthesis, Sound Design, Signal Processing, and Programming*. Cambridge, MA: The MIT Press.

[Di Scipio 2000] Di Scipio, A. (2000). « An Analysis of Jean-Claude Risset's Contours » in : JNMR, 29(1), 1-21

[Dodge et Jerse 1985] Dodge, C., Jerse, T.A. (1985). *Computer Music: Synthesis, Composition, and Performance*. New York: Schirmer Books.

[Lorrain 1980] Lorrain, D. (1980). *Analyse de la bande magnétique de l'oeuvre de Jean-Claude Risset "Inharmonique"*. Paris : Centre Georges Pompidou (Rapport IRCAM n° 26/80).

[Mathews 1969] Mathews, Max V. et al. (1969). *The Technology of Computer Music*. Cambridge: The M.I.T. Press.

[Risset 1969] Risset, J.-C. (1969). *An introductory catalog of computer-synthesized sounds.* Reprinted with C.D. Wergo 2033-2, *The historical CD of digital sound synthesis* (1995), 109-254.

[Risset 1995] Risset, J.-C. (1995). *My 1969 Sound Catalogue : Looking Back from 1992*. in C.D. Wergo 2033-2, *The historical CD of digital sound synthesis* (1995), 88-108.

[Risset 2002] Risset, J.-C., Arfib, D., de Sousa Dias, A., Lorrain, D., Pottier, L. (2002). « De "Inharmonique" à "Resonant Sound Spaces" : temps réel et mise en espace » in *Actes des neuvièmes Journées d'Informatique Musicale*, Marseille : ADERIM-GMEM, 83-88.

[Vercoe 1986] Vercoe, B. (1986). *CSOUND: A Manual for the Audio Processing System and Supporting Programs*. Cambridge MA: MIT Media Laboratory.