

Mémoire de Master 2 Recherche

Construction, Communication et Appropriation des Savoirs

Scientifiques et Techniques

Mention — Histoire, Philosophie et Didactique des Sciences

Parcours — Histoire et Philosophie des Sciences

L'informatique dans la musique de Karlheinz Essl

Romain Versaevel, M2 HPDS, Université Lyon 1

dirigé par Jean-Marc Wolff (Lycée Henri IV)

et Jonathan Simon (Université Lyon 1)

Jury : Hugues Chabot (Université Lyon 1),

Gilles Aldon (Institut Français de l'Éducation),

Violaine Prince (Université Montpellier 2, rapporteur)

30 août 2016

Table des matières

1	Introduction	3
2	Parcours de Karlheinz Essl	4
2.1	Formation : entre rock et éducation classique	4
2.2	Vers la musique algorithmique	9
2.3	IRCAM et Composition en temps réel	16
2.4	Descente de la tour d'ivoire	21
2.5	Conclusion	27
3	Le rôle de l'informatique à travers ses programmes	28
3.1	Introduction	28
3.2	La <i>Realtime Composition Library</i>	28
3.2.1	Description	29
3.2.2	Analyse	34
3.3	La <i>Lexikon-Sonate</i>	40
3.3.1	Description	41
3.3.2	Étude de deux modules : TRILLER et MELOCHORD	45
3.3.3	Analyse	51
3.4	Aspects matériels du « hasard » et du « temps réel »	56
3.4.1	Le « hasard »	57
3.4.2	Le « temps réel »	59
3.5	Un compositeur de l'ère numérique	62
3.6	Conclusion : Essl et l'ordinateur	67
4	Conclusion	68
5	Bibliographie	72

1 Introduction

Dans la deuxième moitié du XX^e siècle, l'informatique moderne, qui était née dans les sphères militaire et académique, a progressivement conquis et révolutionné de nombreux autres domaines d'application. À la fois science centrée sur le calcul et la logique, et technologie dont les principaux représentants sont l'ordinateur et le réseau, elle est aujourd'hui entrée dans les mœurs, conduisant à l'apparition d'une « culture numérique » mondiale. Si bien que, comme le souligne l'historien Philippe Breton, « la question qui est aujourd'hui d'actualité en matière d'informatique est celle de la maîtrise des enjeux que soulève son insertion dans la vie quotidienne »¹ [Breton, 1987, avant-propos]. Le champ artistique, qui a toujours exploité les ressources de la technologie, ne fait pas exception dans le processus d'assimilation de l'informatique. L'ordinateur a bouleversé les formes et les disciplines — réalité augmentée, dispositifs multimédias, œuvres interactives... —, conduisant au développement de l'*art numérique*.

L'objet de ce mémoire est d'étudier cet investissement de l'art contemporain par l'informatique, à travers le cas du compositeur contemporain autrichien Karlheinz Essl. Dans son œuvre, Essl a en effet assimilé les technologies numériques et se les est appropriées en leur donnant une place de premier plan, ce qui fait de ses travaux un exemple particulièrement intéressant d'interface entre art et technique informatique. Il s'est ainsi spécialisé dans la musique algorithmique, c'est-à-dire le recours à des méthodes formelles, généralement appliquées par un programme informatique, pour la composition. Puis, grâce aux progrès technologiques entraînant la diminution des temps de calcul, Essl a conjugué ces méthodes de composition informatiquement automatisée avec des dispositifs de production de son en temps réel, élaborant ainsi la notion de « *composition en temps réel* ».

Notre démarche sera de faire l'étude historique de ce compositeur et de sa pratique. Le code des programmes écrits par Essl est la principale source sur laquelle reposent nos analyses. À travers ces programmes, nous montrerons que les innovations conceptuelles réalisées par Essl, qui représentent de véritables apports qualitatifs sur le plan artistique, l'ont été dès le moment où le progrès technologique incrémental les a ren-

1. Pour un exemple d'étude approfondie et récente du rôle joué par l'informatique dans divers secteurs, voir [Cortada, 2004].

dues possibles.

Essl a fait l'objet de plusieurs travaux biographiques², et ses œuvres sont régulièrement décrites, commentées et analysées³. Cependant aucun de ces travaux n'a l'ampleur de celui-ci. En outre, ils se concentrent sur les aspects artistique et esthétique. L'originalité du présent document est d'approcher Essl comme un acteur historique de la révolution numérique, et sa production comme source au-delà de sa qualité artistique.

Une première partie décrit le parcours du compositeur, en montrant le parallèle avec l'expansion du rôle de l'informatique dans la société, et la construction des notions importantes de son œuvre. La seconde partie, grâce à l'étude en détail des programmes d'Essl — sa bibliothèque *Realtime Composition Library* (1993-) et sa pièce la plus renommée, la *Lexikon-Sonate* (1992-2007) — interroge ces notions, et montre en quoi l'ordinateur est un outil dans son processus créatif.

2 Parcours de Karlheinz Essl

Nous distinguons quatre parties dans le parcours de Karlheinz Essl, dont un portrait est visible sur la figure 1. D'abord ses premiers pas de musicien et ses études, qui voient la formation des influences qui marqueront tout son œuvre. Ensuite, l'assimilation de la *musique algorithmique* qui coïncide avec ses premières utilisations de l'ordinateur, puis le bouleversement de cette démarche provoqué par la rencontre avec le *temps réel*. Enfin, le mouvement du compositeur vers le performeur, marqué par le recours à l'improvisation et aux pratiques artistiques collaboratives.

2.1 Formation : entre rock et éducation classique

Né le 15 août 1960 à Vienne, Karlheinz Essl est issu d'un milieu aisé et cultivé. Son père, Karlheinz Essl senior, est un entrepreneur, qui a fait fortune en fondant la chaîne

2. En particulier [Ehrler, 1999], [Felber, 2007], [Essl et Kühnelt, 2006/2007], [Dobretsberger, 2006], [Sinkovicz, 2005], [Klein, 2013].

3. Par exemple dans [Collins, 2010, chap. 8] ou [Rowe, 2004, p. 306-308].



FIGURE 1 – Portrait de Karlheinz Essl en 2009. Photo © Johannes Tichy

de magasins de bricolage bauMax⁴. Il est aussi connu comme collectionneur d'art, ayant acquis avec sa femme Agnes de nombreuses œuvres contemporaines à partir des années 1970, collection qui l'a conduit à faire construire son propre musée, le *Essl Museum*, dans la ville de Klosterneuburg située sur le Danube à quelques kilomètres au nord de Vienne.

Karlheinz junior est ainsi au contact d'un environnement culturel savant⁵ dès son plus jeune âge. Il étudie le piano à partir de l'âge de sept ans. Il explique cependant que cette expérience lui a déplu et qu'il n'a vraiment apprécié la musique que lorsque, adolescent, il s'est intéressé à la musique rock [Ehrler, 1999]. Il prend alors part à plusieurs groupes et s'initie, en autodidacte, à la guitare électrique puis à la contrebasse. Néanmoins, ce n'est pas le rock « grand public » qui attire Essl, qui cite plus volontiers des groupes avant-gardistes ou expérimentaux comme *Gentle Giant* ou *Can*. C'est à travers ce dernier qu'Essl découvre la musique du compositeur allemand Karlheinz Stockhausen (1928-2007) (deux membres de *Can*, le bassiste Holger Czukay et le claviériste Irmin Schmidt ont tous deux été ses élèves). Âgé de 15 ans, il achète un vinyle

4. Entreprise dissoute en 2015, suite aux conséquences de la crise financière de 2007.

5. On parle de « musique savante » pour désigner la tradition musicale occidentale fortement structurée et théorisée, qui est généralement opposée aux musiques traditionnelle et populaire. Dans la suite de ce document on préférera ce terme à celui de « musique classique », fréquemment employé mais prêtant à confusion (cette musique « classique » n'étant pas limitée à la période classique (1750-1820)).

de Stockhausen, découverte qu'il décrit comme un véritable « choc »⁶. Cet engouement naissant pour la musique électronique le conduira même à construire un petit synthétiseur [Essl et Kühnelt, 2006/2007].

Lorsqu'il arrive dans les études supérieures, Essl, bien qu'ayant suivi une formation en chimie⁷ décide de se consacrer à la musique. Cette décision est prise à l'encontre de ses parents, qui le destinaient à prendre la succession de la direction de l'entreprise familiale (ce que fera finalement son frère cadet, Martin Essl). Reçu à la prestigieuse Académie de musique et des arts du spectacle de Vienne (*Universität für Musik und darstellende Kunst Wien*), il y étudie l'harmonie et le contrepoint avec Alfred Uhl (1909-1992), la contrebasse avec Heinrich Schneikart (1929-2008), et la composition auprès de Friedrich Cerha (1926-). Progressivement il va abandonner la contrebasse pour se consacrer exclusivement à l'analyse musicale et à la composition, suivant en outre des cours de Dieter Kaufmann sur la musique électro-acoustique. Ce parcours académique le conduit à rédiger une thèse sur la « pensée-synthèse » chez Anton Webern sous la direction du musicologue Hans Schneider, achevée en 1991 [Essl, 1991]. L'influence de la musique de Webern, et plus généralement de la musique serielle (voir ci-après), traverse l'ensemble de l'œuvre d'Essl. Il a notamment rendu hommage à Webern à l'occasion du soixantième anniversaire de sa mort, à travers un morceau et un programme (*WebernSpielWerk* et *WebernUhrWerk*, 2005).

Au cours de ses études, Essl fait deux rencontres décisives. D'abord, en 1985, il rend visite à son ami Gerhard Eckel en stage à l'Institut de Sonologie (*Instituut voor Sonologie*) de La Haye (Pays-Bas), l'un des premiers centres de recherche en musique électronique et musique informatique. C'est à cette occasion qu'il fait la connaissance de Gottfried Michael Koenig (1926-), compositeur allemand pionnier de la musique algorithmique, qui dirige l'Institut depuis 1964. Essl trouve la partition d'un quatuor écrit

6. « À la maison, j'ai mis mon casque et écouté le vinyle. J'étais complètement sous le choc. À l'époque, je trouvais cette musique atroce, qui pourtant m'a emballé. » [Essl et Hötzenecker, 2016] (« *Zuhause habe ich mir Kopfhörer aufgesetzt und mir die Platte angehört. Ich war total schockiert. Das war für mich damals ganz grauenhafte Musik, aber es hat mich gepackt.* »). Dans la suite de ce document, toutes les citations traduites par nos soins sont transcrrites en note de bas de page dans leur langue originale.

7. *Oberstufe Gymnasium*, parcours du système scolaire allemand et autrichien équivalent au lycée (*Gymnasium*), mais plus long d'un an, ce qui permet d'approfondir une discipline de spécialité.

par Koenig (*Streichquartett 1959*, 1959) et, ayant des difficultés à l'analyser, contacte celui-ci, qui lui apprend qu'il a utilisé des opérations de hasard dans sa composition. Les deux compositeurs correspondent et collaborent depuis, Essl ayant même été invité à contribuer à l'élaboration de *Projekt 3*, projet resté inachevé faisant suite aux *Projekt 1* (1964) et *Projekt 2* (1966), premiers programmes pour la Composition Assistée par Ordinateur (CAO). Puis, en 1988, Essl rencontre John Cage (1912-1992), compositeur américain emblématique du XX^e siècle. Il connaît déjà son œuvre mais peut l'approcher en personne lors d'un concert donné en son honneur à Vienne. L'œuvre de Cage marque profondément celle d'Essl, qui lui rend d'ailleurs hommage dans deux de ses créations : *In The Cage* (1987) et *FontanaMixer* (2004).

Cette période de formation, qui s'achève avec la soutenance de thèse en 1991, voit se former le style d'Essl. On peut y identifier ses principales influences théoriques.

D'abord, il y a le **sérialisme**, auquel l'a initié son professeur Friedrich Cerha⁸. Ce mouvement, fondé à Vienne au début du XX^e siècle par la seconde école de Vienne (Arnold Schönberg, Alban Berg et Anton Webern), élabore à l'origine des techniques d'inspiration arithmétique pour éradiquer systématiquement la tonalité, le langage musical utilisé dans toute la musique savante occidentale depuis le XVIII^e siècle, qui limite le matériau employé dans chaque composition (par exemple, la tonalité de do majeur n'utilise que les touches blanches du piano). L'objet central du sérialisme est la *série*, qui est un ordonnancement exhaustif du matériau disponible. La série dodécaphonique fait ainsi se succéder les douze hauteurs de la gamme chromatique sans en répéter aucune, évitant ainsi toute ressemblance avec une tonalité identifiable ; des séries peuvent aussi être utilisées pour contrôler les volumes, les rythmes, etc.⁹ De très nombreux compositeurs du XX^e siècle se réclament du sérialisme et l'ont développé, parmi lesquels Stockhausen et Koenig, mais aussi Pierre Boulez (1925-2016), Luigi Nono (1924-1990), Luciano Berio (1925-2003)... Sur le plan esthétique, l'école serielle va de pair avec la philosophie d'Adorno, lui-même compositeur, qui promeut la **Nouvelle Musique** (*Neue Musik*) et l'*injonction avant-gardiste*. Ce précepte esthé-

8. Dans [Essl et Pagano, 2009], Essl précise qu'avant de commencer sa formation auprès de Cerha ses goûts le portaient plutôt vers des mouvances opposées comme le néo-classicisme.

9. La présente étude ne nécessite pas une connaissance pointue du sérialisme sur le plan esthétique. Il faut essentiellement en retenir la volonté de contrôler la musique par des procédés mathématiques. Le lecteur qui souhaiterait en savoir plus est invité à consulter par exemple [Wodon, 2014].

tique, présent dans toutes les disciplines de l'art contemporain, exige des artistes une innovation radicale permanente pour mériter ce statut¹⁰. Elle se manifeste dans l'intérêt qu'Essl portera à l'ordinateur et sa participation à l'avant-garde.

Les autres influences notables sont celles des trois compositeurs déjà cités : **Stockhausen**, **Koenig** et **Cage**. Tous trois ont été parmi les premiers musiciens à explorer les possibilités de l'*œuvre ouverte*¹¹ et du hasard, problématiques très présentes dans l'œuvre d'Essl. Koenig a même affirmé même que « la maîtrise du hasard par le compositeur est un problème central de la musique actuelle »¹² [Koenig, 1962, p. 813]. Le hasard joue plus généralement un rôle essentiel dans l'art du XX^e siècle, qui cherche notamment à s'approprier les notions issues de la théorie du chaos et de la physique quantique. L'influence de Stockhausen apparaît dans l'intérêt porté par Essl au son et en particulier au son de synthèse, avec l'intégration de la musique électronique dans la musique savante. Celle de Koenig est une paternité technique directe, puisqu'il a initié Essl à la musique algorithmique. Celle de Cage réside dans la recherche de formes radicalement nouvelles, en particulier l'événement, la performance. Le compositeur Andrew Gerzso résume ainsi le terreau dans lequel Essl a pu développer son œuvre : « Schönberg a porté un coup terrible à la tonalité et Cage à ce qui fait l'essence d'une œuvre — ouvrant la voie à la notion de *work-in-progress* ou encore d'*œuvre ouverte* »¹³ [Assayag et al., 2009, p. 1].

Enfin, la dernière influence notable d'Essl est d'ordre philosophique ; il s'agit du *constructivisme radical*, auquel le compositeur a même dédié un essai en 1992 [Essl, 1992]. Cette épistémologie repose sur l'idée que la « connaissance » est une construction de l'esprit par rapport au monde, qui peut de fait varier d'un individu à l'autre, sans qu'il soit possible de parler de « vérité ». Essl résume ainsi l'application de ce courant de pen-

10. « Le paradigme adornien d'une négativité en acte dans l'œuvre d'art se lit comme une injonction pour l'art moderne, et en partie pour l'art contemporain, d'avoir à intégrer l'action impérative de l'avant-garde. L'art se doit d'être critique — traduisons : avant-gardiste — et cet impératif s'ajoute à ceux déjà institués par les fondations précédentes. » [Cauquelin, 2010, p. 59]. Pour plus de détails sur la philosophie esthétique d'Adorno, voir [Adorno, 1949] et [Adorno, 1970].

11. Telle que définie par Umberto Eco dans *L'Œuvre ouverte* [Eco, 1962]. Cette notion est décrite plus en détail dans la section 3.3.3.

12. « ... die kompositorische Beherrschung des Zufalls ein zentrales Problem heutigen Komponierens... »

13. « Schönberg dealt a blow to tonality and Cage to the notion of what constitutes a work — opening the way to the notion of work in progress or open-ended works. »

sée à sa pratique artistique : « La musique ne se produit donc pas seulement dans la salle de concert, mais principalement dans la tête de l'auditeur. Savoir que le monde est construit par les perceptions individuelles plutôt que la simple illustration d'une réalité extérieure est lourd de conséquences pour la composition musicale. Cela fait en effet de l'auditeur un co-créateur. »¹⁴ [Essl, 2008]

Bien que le rock ait joué un rôle décisif, à la fois en étant la source de la passion qui a décidé Essl à faire des études de musique, et en l'amenant à Stockhausen et à la musique électronique, la formation du compositeur est avant tout classique, académique, savante. Elle est l'occasion pour lui de forger ses premières influences importantes, qui le suivront toute sa carrière, et de manifester une inclination favorable pour la musique algorithmique.

2.2 Vers la musique algorithmique

On l'a vu, la rencontre avec Gottfried Michael Koenig a amené Essl à s'intéresser à la musique algorithmique. Cet intérêt d'abord théorique connaît par la suite de nombreux développements sur le plan pratique, qui se construisent parallèlement à la relation entre Essl et l'ordinateur.

Ici, il convient de préciser ce que signifie ce terme de « musique algorithmique », et, pour commencer, ce qu'est un algorithme. Le mot « algorithme » est forgé à partir du nom du mathématicien perse du IX^e siècle Muhammad al-Khuwārizmī ; il signifie¹⁵ un « ensemble de règles opératoires dont l'application permet de résoudre un problème énoncé au moyen d'un nombre fini d'opérations » [Larousse 2016]. Dans ce sens, une recette de cuisine aussi bien que la méthode pour poser une multiplication sont des algorithmes. La composition algorithmique est alors simplement « la création d'algorithmes dont le produit a des conséquences musicales notables et utiles au compositeur, allant de la production et de l'exploration de matériaux musicaux à la génération

14. « *Musik findet deshalb nicht allein im Konzertsaal, sondern vor allem im Kopf statt. Die Erkenntnis, dass die Welt nicht nur die Abbildung einer äußeren Wirklichkeit ist, sondern erst durch individuelle Wahrnehmungsarbeit konstruiert wird, führt zu radikalen Konsequenzen für die musikalische Komposition, die letztendlich den Hörer zum Mitschöpfer macht.* »

15. Pour une caractérisation plus poussée, qui n'est pas nécessaire ici, se reporter à [Knuth, 1998].

d'œuvres complètes »¹⁶ [Collins, 2010, p. 299].

Au sens large, la musique algorithmique possède donc une histoire séculaire¹⁷. Tous les traités mettant en place des systèmes de règles pour l'organisation d'une composition en participent, dont le plus ancien connu, intitulé *Musica enchiriadis* (d'auteur inconnu), remonte au IX^e siècle. Le contrepoint rigoureux illustré par exemple dans *L'art de la fugue BWV 1080* (1750) de Johann Sebastian Bach a une essence algorithmique, de même que, de manière totalement explicite, les jeux musicaux comme le *müsikalische Würfelspiel K.294d* (jeu de dé musical, 1792) de Mozart ou le jeu de « cartes musiciennes » de la figure 2.

Mais l'essor de la notion d'algorithme se fait conjointement avec l'avènement de l'informatique. Les ordinateurs sont en effet des machines capables d'appliquer des algorithmes de traitement de données et d'information ; ce sont en outre des machines *programmables*, autrement dit non-spécialisées, auxquelles leur utilisateur peut confier *n'importe quelle* tâche algorithmique, pourvu qu'il la formule dans un langage de programmation. Dans le domaine artistique, l'informatique musicale (*computer music*) a exploré très tôt les possibilités offertes par les algorithmes, à la fois pour la composition et l'analyse musicologique de morceaux existants. Dans les années 1960 et 1970, ces deux axes contribuent à la formation de l'informatique musicale (*computer music*) comme une discipline à part entière, avec l'ouverture de nombreux instituts dédiés à travers le monde et d'organes de communication scientifique : *International Computer Music Conference* à partir de 1974, *Computer Music Journal* à partir de 1977. Le traitement informatique du signal sonore (transformation et synthèse) n'est arrivé que plus tard¹⁸. Cela s'explique par le fait que les données audio sont beaucoup plus volumineuses que les données symboliques nécessaires à la composition (des nombres

16. « *Formally speaking, algorithmic composition is the creation of algorithms whose output has clear musical consequences of use to the composer, from the production and exploration of musical materials to the generation of complete works.* »

17. Cette remarque n'est pas seulement anecdotique : elle montre que l'histoire de la musique est imprégnée d'une culture algorithmique, qui explique que l'informatique ait trouvé un écho important dans cette discipline

18. On peut tout de même citer les premières expériences pour faire jouer de la musique par ordinateur, sur le CSIRAC de l'université de Melbourne et le Ferranti Mark I de l'université de Manchester, toutes deux en 1951. Dans les deux cas, un synthétiseur simple jouait des mélodies populaires (*God Save The King, Baa Baa Black Sheep, Colonel Bogey March...*) fournies à l'ordinateur.

pour représenter les hauteurs, durées... des notes), et nécessitent des interfaces de conversion complexes (carte son). La musique électronique possède en outre une avance considérable dans ce domaine, qu'elle explore depuis le début du siècle — le premier instrument électronique de synthèse musicale, le Telharmonium, est construit en 1906. Dans les années 1980, quand Essl entre véritablement en contact avec les ordinateurs, l'informatique commence à rattraper ce retard ; elle permet de réaliser des synthèses et transformations inédites, impossibles avec la seule électronique, mais les calculs nécessitent encore un temps important.

La première partition générée par un ordinateur est l'*Illiac Suite*¹⁹, qui paraît en 1957. Elle est réalisée grâce à un programme conçu par Lejaren Hiller et Leonard Isaacson, sur le superordinateur ILLIAC I, construit à l'université de l'Illinois en 1952. Les travaux se multiplient rapidement par la suite, tant en musique que dans les autres arts, si bien que l'on parle de « révolution algorithmique ». Toujours en 1957, Iannis Xenakis (1922-2001) a recours à un ordinateur qui l'assiste dans des calculs pour un enregistrement de sa pièce *Diamorphoses* ; et en 1964, c'est *Projekt 1* de Koenig qui voit le jour. L'ordinateur permet d'assouvir deux quêtes de l'art de la deuxième moitié du XX^e siècle, le recours au hasard qui désacralise la figure de l'auteur, et les recherches formelles, qui s'expriment dans les algorithmes²⁰. Dans une synthèse sur le sujet rédigée en 2007 [Essl, 2007], Essl décrit plusieurs chemins d'appropriation de l'ordinateur par les initiateurs de la musique algorithmique : musique générative (dont sa propre démarche s'inspire, voir ci-dessous), automatisation de procédés d'écriture comme ceux du sérialisme, et investigation du hasard (comme les « principes aléatoires » de Koenig ou la « musique stochastique » de Xenakis).

C'est en 1984, qu'Essl utilise un ordinateur pour la première fois, par l'intermédiaire d'un ami, testeur pour un journal. L'année suivante, il investit dans un micro-ordinateur Atari ST, à sa sortie. Les années 1970 et 1980 correspondent à un changement radical dans les pratiques informatiques, qu'il illustre cet achat. Alors qu'auparavant la logique de conception d'ordinateurs tendait vers des machines centralisées de plus en plus puissantes, apparaissent les *micro-ordinateurs*, à usage individuel, qui vont progressivement s'imposer au détriment des premières. Le tableau 3 montre quelques

19. Voir [Hiller et Isaacson, 1979].

20. En littérature, ces deux tendances sont par exemple illustrées respectivement par le mouvement surréaliste et par l'OuLiPo.



FIGURE 2 – Deux cartes d'un jeu de « cartes musiciennes » de 1830, conservé à la Bibliothèque Nationale de France (auteur inconnu). Les mesures indiquées en-dessous des cartes sont conçues de manière à ce que le morceau formé par la juxtaposition de n'importe quelle suite ordonnée de cartes (As, Roi, Dame, etc., quelles que soient les couleurs choisies) soit une valse cohérente. Ce jeu forme ainsi un système de règles et a donc une nature algorithmique.

modèles emblématiques de cette tendance²¹, caractérisée par une explosion à la fois des ventes et du nombre de modèles. Quant au tableau 4²², il montre que l'Atari ST était significativement moins cher que la plupart de ses concurrents, ce qui est l'un des facteurs de son succès. Il a aussi été particulièrement populaire parmi les pionniers de la musique informatique du fait de son interface MIDI (norme définie deux ans plus tôt, pour un format de représentation informatique des données musicales) intégrée. L'acquisition de ce micro-ordinateur par Essl s'inscrit donc pleinement dans la tendance historique de cette période. Comme nous le verrons dans les parties suivantes (2.3 et 2.4), l'idéal de liberté associée à cette individualisation de l'ordinateur jouera aussi un rôle important dans la suite du parcours d'Essl. Philippe Breton la décrit ainsi : « L'invention du micro-ordinateur [...] avait pour objectif explicite de battre en brèche la centralisation et la possession des précieuses « informations » par quelques privilégiés. La « guérilla » micro-informatique [...] a été en grande partie à l'origine de la « culture informatique », partagée dans un large public et facteur de démocratisation de la vie sociale et du savoir. » ; il ajoute que « Pour les générations nées dans les années soixante, informatique et liberté sont désormais synonymes. » [Breton, 1987, p. 206].

21. D'après [Reimer, 2005] et <http://www.oldcomputers.net/> (consulté le 30 août 2016).

22. D'après

<https://web.archive.org/web/20091026193531/http://www.geocities.com/SiliconValley/Vista/3015/16bit.html> (consulté le 30 août 2016).

Modèle	Année de lancement	Nombre d'exemplaires vendus
Programma 101	1965	35.000 à 40.000
Altair 8800	1975	Quelque milliers
Apple II	1977	~ 2.000.000
IBM PC	1981	Plusieurs millions
Macintosh	1984	~ 2.000.000
Atari ST	1985	~ 6.000.000
Commodore 64	1982	17.000.000 à 25.000.000

FIGURE 3 – Quelques modèles de la naissance de la micro-informatique.

Modèle	Commodore 64	Atari 1040 ST	Commodore Amiga	Apple Macintosh Plus	IBM PC AT
Prix	595\$	999\$ (mono) ou 1199\$ (couleur)	1795\$	2195\$	4675\$

FIGURE 4 – Prix des micro-ordinateurs les plus répandus en 1985

Sur son Atari ST, Essl programme d'abord en BASIC (*Beginner's All-purpose Symbolic Instruction Code*), un langage de programmation impératif généraliste conçu pour sa simplicité d'utilisation. Il se tourne ensuite vers un langage beaucoup moins répandu, le LOGO²³. Celui-ci a été développé à partir de la fin des années 1960, à l'origine par deux chercheurs du MIT, Seymour Papert et Marvin Minsky. Les programmes écrits en LOGO dirigent le déplacement d'un curseur représentant une tortue ; leur résultat est un dessin, celui de la trajectoire de la tortue. Cette apparence ludique est due au fait que le LOGO, conçu en s'appuyant sur les théories de Jean Piaget, était destiné à permettre à des enfants de se familiariser avec la programmation ; il s'agit cependant d'un langage très complet qui permet par exemple de manipuler des fichiers ou des structures de listes. C'est principalement cette dernière fonctionnalité qui retient l'attention d'Essl, lequel implémente des procédures issues du sérialisme. Essl développe en LOGO une bibliothèque, intitulée *COMPOSE* (1988-), c'est-à-dire un ensemble de fonctions qu'il peut utiliser dans la composition de diverses pièces.

23. Plus précisément le xLOGO, l'une des nombreuses implémentations du langage original. Pour plus d'informations sur le LOGO, voir [Harvey, 1985].

La pensée algorithmique devient prépondérante dans les œuvres du jeune compositeur. Sa première pièce ayant nécessité un programme informatique, *BWV 1007a* paraît en 1986 (en collaboration avec son ami Eckel). Le programme était conçu pour réaliser une analyse statistique de la *Première suite pour violoncelle* de Jean-Sébastien Bach (BWV 1007, 1720), afin de la découper et recombiner ensuite, de sorte que cette recombinaison soit mélodiquement cohérente. Précisons que cette expression peut être trompeuse : lorsque l'on parle de programmes et d'algorithmes, il est facile de laisser imaginer une « volonté » ou une forme de conscience de l'ordinateur. Il n'en est évidemment rien, le programme d'Essl ne fait en réalité qu'exécuter des instructions prévues par lui. Une recombinaison « mélodiquement cohérente » signifie ainsi simplement une recombinaison respectant des règles définies par Essl dans le but d'éviter des combinaisons sonores qu'il juge incohérentes. En cela, *BWV 1007a* est un bon exemple de ce qu'est — et donc de ce que n'est pas — un algorithme : Essl aurait théoriquement pu appliquer lui-même sa méthode, découper et recombiner la pièce de Bach, en s'assurant de respecter les règles mélodiques qu'il a prévues ; mais cette tâche aurait été particulièrement fastidieuse ; l'ordinateur permet de l'automatiser et de la réaliser plus rapidement. La même année, Essl compose l'une de ses premières œuvres majeures, le quatuor à corde *Helix 1.0*, qui lui vaut deux prix en 1987, au concours international de quatuors à cordes de Budapest et au concours de quatuors à cordes du Konzerthaus de Vienne. Cette pièce construite sur des structures élaborées représentant des mouvements en hélice (comme combinaison de deux mouvements, l'un circulaire, l'autre rectiligne) reflète une véritable pensée algorithmique²⁴. Les algorithmes impliqués ont cependant été réalisés sans l'aide de l'ordinateur.

Il faut noter que l'attraction pour la musique algorithmique est totalement emblématique de cette période. Tous les compositeurs ne s'intéressent pas à l'informatique — l'une des caractéristiques de l'art depuis le XX^e siècle est le nombre et la pluralité des écoles concurrentes — mais c'est une tendance forte dans le milieu musical. Tendance marquée par le développement d'*environnements informatiques de composition*, des logiciels, dont certains offrent les fonctionnalités de langages de programmation graphiques, qui simplifient la tâche de conception algorithmique des compositeurs. Ces

24. Encore une fois, il y a deux causes à cette présence structurante forte des algorithmes : l'influence de la programmation informatique qui occupe Essl à cette période, mais aussi l'omniprésence de procédés d'ordre algorithmique dans la tradition musicale.

environnements se multiplient à partir du milieu des années 1980 — et leur amélioration ainsi que le développement de nouveaux environnements se poursuit jusqu'à aujourd'hui — : *Autobusk* en 1986, *Music Box* en 1986, *HMSL (Hierarchical Music Specification Language)* et *Voyager* en 1987, *Finale* en 1988, *Cubase* et *Common* en 1989... .

Pour un compositeur qui a accordé aux aspects algorithmiques de la composition une place prépondérante dans son œuvre, Essl a une relation inhabituelle aux algorithmes. Sa position est résumée par cet échange dans une interview de 2015 : « Magdalena Halay : Qu'est-ce qui pour vous fait qu'une composition sonne algorithmique ? — Karl-heinz Essl : Oh, le but n'est pas que mes compositions sonnent algorithmiques ! [...] Le but est qu'elles ne sonnent pas algorithmiques. »²⁵ [Essl et Halay, 2015]. Les méthodes formelles impliquées dans le processus de composition d'Essl sont *théorisées, décrites* (dans des articles, des programmes accompagnant les performances ou leurs enregistrements, les cours et les interviews donnés par Essl...) ; mais elles ne sont pas *audibles*, ou du moins ne sont pas mises au premier plan lors de l'écoute. Cette attitude s'oppose à celle de beaucoup d'artistes contemporains ayant recours à des algorithmes, qui au contraire en font la monstration à travers leurs œuvres²⁶. Elle opère ainsi une sorte de synthèse entre l'esthétique de la virtuosité (du compositeur) et celle qui met les moyens de composition au service de l'expérience de l'auditeur.

Ainsi, les dernières années d'études d'Essl, qui sont aussi ses premières années en tant que compositeur, marquent ainsi l'apparition et le développement dans sa pensée artistique de pratiques algorithmiques. Ce phénomène va de pair avec celui de l'avènement de la micro-informatique, qui permet à Essl d'expérimenter avec sa propre machine. Il élabore alors une articulation relativement originale entre le procédé algorithmique et son résultat esthétique. Comme nous allons le voir dans la partie suivante, les progrès technologiques vont lui permettre d'enrichir encore grandement cette originalité.

25. « MH: What makes an algorithmic composition sound algorithmic to you? — KHE: Oh, it's not the point that it sounds algorithmic! Maybe this is a big misunderstanding [laughs]. The point is that it must not sound "algorithmic", you know? [laughs] »

26. On peut trouver de nombreux exemples dans [Wilson, 2012], en particulier au chapitre 7 intitulé « *Algorithms* », p. 160-179.

2.3 IRCAM et Composition en temps réel

Un tournant décisif dans l'histoire de la musique informatisée et dans la carrière d'Essl a lieu à la fin des années 1980 et au début des années 1990. Il s'agit de l'apparition de dispositifs en « temps réel »²⁷. Ceux-ci offrent pour la première fois la possibilité d'une interaction immédiate entre l'utilisateur et une machine *informatique*²⁸ qui convertit le résultat de ses calculs en événements musicaux significatifs, typiquement sous forme de sortie MIDI (ce qui permet de produire du son avec des instruments comme le Yamaha Disklavier²⁹) ou directement dans un format audio (écouté *via* des haut-parleurs).

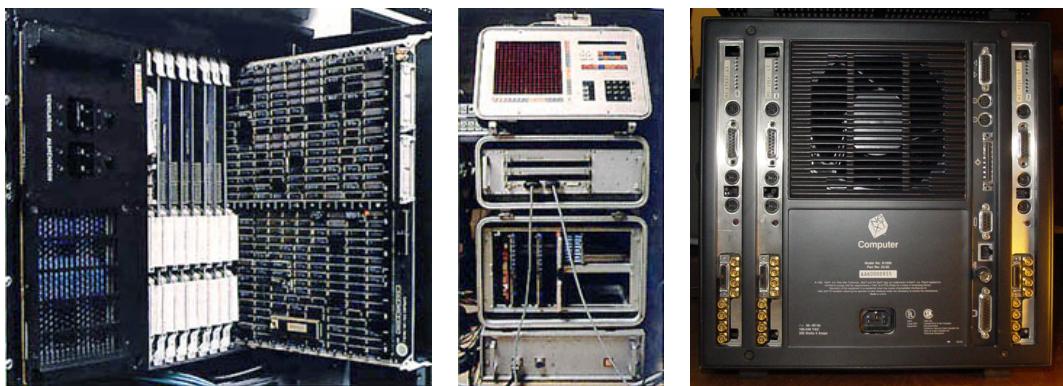


FIGURE 5 – Trois photographies de composants de la *Station d'informatique musicale* de l'IRCAM. Il s'agissait d'un dispositif reliant plusieurs machines (ordinateurs, cartes son, claviers, interfaces) qui occupait une pièce entière. © Jean-Bernard Emond.

Essl est confronté avec ces dispositifs en temps réel pour la première fois en 1992. Il est alors invité à l'Institut de recherche et coordination acoustique/musique (IRCAM) à Paris, institut public de création musicale et de recherche appliquée à la musique, qui lui passe commande d'une pièce pour ensemble et sa toute récente *Station d'informatique musicale* (SIM)³⁰, visible sur la figure 5. Cette machine succède à plusieurs autres prototypes développés par l'ingénieur et physicien italien Giuseppe di Giugno

27. Ce que recouvre exactement la notion de temps réel en musique est discuté plus précisément dans la partie 3.4.2.

28. La nouveauté réside dans le fait que c'est avec un ordinateur, une *machine à calculer*, que le temps réel est obtenu : en musique électronique, les synthétiseurs permettaient ce temps réel depuis plusieurs décennies... et les instruments de musique traditionnels, depuis toujours.

29. Piano mécanisé créé en 1987 ; la norme MIDI a quant à elle été établie en 1983.

30. Voir [Lippe et al., 1991] pour une description détaillée de la station.

depuis 1975, intitulés 4A, 4B, 4C et 4X. Essl non seulement compose la pièce qui lui a été commandée (*Entsagung*, 1991-1993), mais s'initie aussi au langage de programmation MAX avec lequel fonctionne la SIM, tirant profit de sa familiarité avec la programmation acquise au contact du LOGO pour réaliser des expériences de composition. Le « temps réel » de l'IRCAM entraîne naturellement un gain de temps considérable, à travers deux ruptures. D'une part, les calculs sont exécutés plus rapidement : alors que les résultats des algorithmes de la bibliothèque *COMPOSE* nécessitaient parfois une nuit entière de calcul, la SIM les calcule immédiatement. D'autre part, ces résultats sont audibles, c'est-à-dire dans un format directement intelligible ; lorsqu'il programmait en LOGO, Essl n'obtenait que des résultats sous forme symbolique (par exemple des nombres représentant des hauteurs de notes et leurs durées), qu'il devait ensuite traduire en partition et interpréter sur un instrument traditionnel.

La découverte des machines en « temps réel » et de MAX est déterminante pour Essl, qui les adopte immédiatement comme des outils de travail de premier plan. Il transporte dans ce nouveau langage les algorithmes de *COMPOSE* dans une bibliothèque intitulée *Realtime Composition Library (RTC-lib)*, qu'il continue à enrichir et à optimiser. Il compose plusieurs pièces interactives en temps réel, parmi lesquelles son œuvre la plus célèbre, la *Lexikon-Sonate*. La *RTC-lib* et la *Lexikon-Sonate* font l'objet respectivement des sections 3.2 et 3.3.

L'interaction musicale en temps réel n'était cependant possible au début des années 1990 qu'avec les machines prototypiques conçues par des centres de recherche comme l'IRCAM, qui n'existaient donc qu'en nombre très limité. L'accès y était de fait difficile et concurrentiel ; en outre, Essl souhaitait composer en étant autonome de telles institutions. Il lui a fallu pour cela attendre 1998 et la sortie du premier iMac, suffisamment puissant pour obtenir des performances similaires à celle de la Station d'informatique musicale. Comme le montre la figure 6, cela coïncide avec un forte augmentation de sa productivité (en tant que nombre d'œuvres publiées).

Ces œuvres d'une forme inédite sont le volet pratique du développement d'Essl, qui formule aussi de nouvelles notions sur le plan théorique, le conduisant à son rapport actuel à la musique algorithmique. Il élabore en particulier deux notions étroitement liées : celle de « composition en temps réel » (*Echtzeit Komposition*) et celle de « *Struk-*

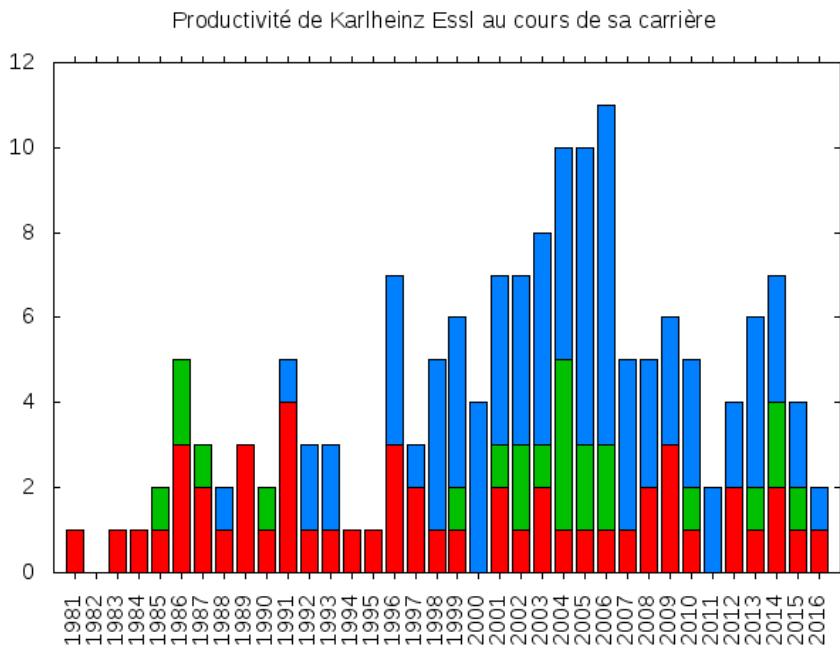


FIGURE 6 – Graphique présentant le nombre de pièces produites par Karlheinz Essl par année. Les œuvres reportées en rouge sont des morceaux « classiques », celles en vert utilisent de la musique électronique mais seulement sous forme enregistrée, celles en bleu ont recours à la l'informatique pour leur exécution.

turgenerator »³¹.

Voici, pour les appréhender, deux définitions qu'Essl donne aujourd'hui des algorithmes : « [Les algorithmes] sont le fondement de ma pensée musicale, qui consiste à ne pas voir la musique seulement comme une expérience sensible, mais comme quelque chose qui comporte une multitude de structures plus profondes que l'on peut exprimer sous forme de modèles. »³² [Essl et Hötzenecker, 2016], et : « Il y a la définition classique, qui provient plutôt des sciences de l'ingénieur, selon laquelle un algorithme est une sorte de recette de cuisine pour résoudre rapidement un problème. C'est une approche possible, mais il y en a une autre que je trouve plus intéressante. Elle consiste à envisager l'algorithme comme la définition d'un méta-modèle, duquel on peut obtenir différents résultats en modifiant les paramètres du système. C'est en ce sens que j'utilise le terme d'algorithme, et c'est ainsi que je le comprends dans les travaux de

31. Littéralement, « générateur de structure » ; dans ce qui suit nous conserverons le mot original.

32. « Diese sind auch die Grundlage meines Musikdenkens, Musik nicht nur als ein sinnliches Erlebnis zu sehen, sondern als etwas, das viele Tiefendimensionen hat, die sich in Form von Modellen ausdrücken lassen. »

Gottfried Michael Koenig et Karlheinz Stockhausen. »³³ (cité par [Förster, 2011]).

Essl fait aussi plusieurs fois référence à la notion d'*Urpflanze* de Goethe, une plante originelle imaginaire dont on pourrait dériver toutes les espèces de plantes [Goethe et al., 1993].

Le mot de *modèle* est crucial dans l'idée de *Strukturgenerator* [Essl, 1996]. Essl conçoit des algorithmes dont le produit peut varier selon les paramètres fournis en entrée et le hasard³⁴ interne des opérations exécutées par l'ordinateur, mais tout en respectant une structure imposée commune. Le module *Triller* de la *Lexikon-Sonate*, étudié dans la partie 3.3.2, est un exemple simple de *Strukturgenerator*. Tout ce que produit ce programme a une *structure* de trille : un ensemble de quelques notes répétées à grande vitesse ; mais les trilles générées par différentes utilisations varient par le nombre de notes différentes jouées, leurs hauteurs, la vitesse, la durée du trille... L'idée de définir des éléments musicaux comme des représentants d'un ensemble des possibles gouverné par une structure, sous la forme d'un système de règles, n'est toutefois ni nouvelle ni originale. Les règles strictes gouvernant le contrepoint classique relèvent exactement de ce cas de figure ; la seule nouveauté dans les *Strukturgeneratoren* d'Essl est qu'ils sont *actifs*, l'ordinateur permettant de créer la musique à la demande une fois le système de règles explicité et traduit en code informatique. On retrouve des procédés similaires dans nombre de musiques algorithmiques où le temps réel n'est pas nécessaire, et beaucoup de domaines usent de tels schémas, par exemple dans les jeux vidéos où il est aujourd'hui monnaie courante de générer ainsi les décors.

En revanche, l'originalité d'Essl réside dans la manière d'utiliser ces *Strukturgeneratoren* : ce qu'il nomme la « composition en temps réel ». Celle-ci peut être décrite comme la conjugaison de la composition algorithmique avec le temps réel. Les pièces de l'œuvre d'Essl répondant à cette appellation peuvent être réparties entre deux catégories : les *installations* et les *méta-instruments*. Les installations sont des programmes qui génèrent de la musique sans intervention humaine, généralement mises en place dans un lieu précis comme à l'intérieur d'un musée. À la différence d'un simple enre-

33. « *Da gibt es klassische Definitionen, die eher aus den Ingenieurwissenschaften kommen, die beschreiben den Algorithmus als eine Art Kochrezept um schnell zu einer Lösung zu kommen. Dies ist ein möglicher Ansatz, aber es gibt noch einen weiteren, den ich interessanter finde. Dort begreift man einen Algorithmus mehr als Definition eines Metamodells, aus dem heraus durch Veränderung der Systemparameter verschiedenste Resultate entstehen. Diesen Algorithmusbegriff verwende ich selber und finde ihn auch in Werken und Arbeiten von Gottfried Michael Koenig und Karlheinz Stockhausen.* »

34. Le rôle du hasard dans la composition en temps réel fait l'objet de la partie 3.4.

gistrement, le son émis par un programme de type installation est élaboré au fur et à mesure de sa diffusion. Ses caractéristiques notables sont en particulier l'absence de répétition, mais aussi de début et de fin en tant que tels. Les *méta-instruments* sont des programmes offrant une interaction limitée (avec un *instrumentiste*), mais dont le produit est du même type que celui d'une installation ; ce sont en quelque sorte des installations interactives dont un petit nombre de paramètres internes peut être modifié en temps réel. Un instrument traditionnel comme le violon laisse une liberté à celui qui en joue (hauteur, durée, volume des notes...), mais impose aussi certains paramètres du son (timbre, ambitus, nombre de notes pouvant être jouées simultanément...). C'est ce qui justifie cette appellation de méta-instrument, la principale différence étant que les instruments traditionnels sont déterministes (le même geste produira systématiquement le même son) mais généralement pas les méta-instruments programmés par Essl. Comme nous le verrons par exemple à travers l'étude détaillée de la *Lexikon-Sonate*, c'est en combinant et associant des *Strukturgeneratoren* qu'Essl réalise de tels programmes.

La spécificité de la composition en temps réel réside dans cette rencontre à la fois de la composition algorithmique et du temps réel, et représente une rupture qualitative par rapport à l'une et à l'autre. En effet, l'utilisation d'algorithmes n'était auparavant possible qu'en amont du moment de réalisation de la pièce. Quant aux technologies en temps réel, celles qui existaient déjà en musique électronique ne pouvaient être utilisées qu'en tant qu'instruments, pour produire et transformer du son. Les œuvres créées avec les prototypes d'ordinateurs en temps réel s'inscrivaient dans cette continuité, en utilisant la puissance de calcul pour élaborer de nouvelles transformations du son (par exemple *Répons* (1981-1984) de Pierre Boulez ou *Antara* (1985-1987) de George Benjamin). Essl a été l'un des premiers à mettre ces machines à calculer au service d'algorithmes de composition, et sans doute celui parmi ces pionniers qui a le plus approfondi cette notion. Par la suite, avec la démocratisation de l'informatique (tant matérielle que logicielle), les pratiques similaires se sont multipliées, avec par exemple l'apparition au début du XXI^e siècle du *live coding* (utilisation d'environnements de programmation interactifs pour « improviser » des codes, écrivant ainsi devant une audience le code informatique générant la pièce entendue en même temps).

Cette période est donc celle de la maturité pour Essl, celle où il affirme son style et

définit son originalité. En découvrant le traitement informatique de la musique « en temps réel », et en s'en servant pour élargir la portée de ses travaux algorithmiques plutôt que de se concentrer sur le traitement du signal, il met au point la notion de *composition en temps réel* qui sera sa marque de fabrique. C'est aussi la période où il prend totalement place dans le monde de la musique contemporaine. Ses études finies, il est invité à enseigner aux prestigieux *Darmstädter Ferienkurse für Neue Musik* (cours d'été de Darmstadt pour la Nouvelle Musique) de 1990 à 1994, auxquels ont participé beaucoup des grands compositeurs du XX^e siècle, en particulier germanophones (Adorno, Boulez, Cage, Stockhausen, Xenakis, mais aussi Luciano Berio (1925-2003), György Ligeti (1923-2006), Olivier Messiaen (1908-1992), Edgar Varèse (1883-1965)...); il devient ensuite professeur à l'université Anton Bruckner de Linz et donne des conférences dans de nombreuses universités à travers le monde.

2.4 Descente de la tour d'ivoire

Le dernier événement décisif dans le parcours d'Essl a lieu en 1997, lorsqu'il est invité au festival de Salzbourg (*Salzburger Festspiele*) et présenté parmi les « compositeurs de la nouvelle génération ». Le festival de Salzbourg, dédié à l'opéra, au théâtre et à la musique classique, est un événement de premier plan dans ces domaines. Il est de plus davantage tourné vers la musique historique (en particulier celle de Mozart, né à Salzbourg) que vers la création contemporaine ; l'invitation faite à Essl marque donc un sommet de sa notoriété de compositeur.

Cet aboutissement est cependant un coup d'arrêt pour Essl, que ces honneurs conduisent à prendre du recul et à interroger son parcours. Il décide alors de changer radicalement sa posture de compositeur, en quittant la « tour d'ivoire » dans laquelle il travaillait jusqu'alors. Au lieu de seulement composer seul des morceaux (ou des programmes) et d'en confier l'exécution à des interprètes (ou des machines), il souhaite s'ouvrir vers le public et le reste du milieu artistique. Cela le conduit d'une part à devenir *compositeur-performeur*, d'autre part à chercher à collaborer avec d'autres artistes pour ses créations futures.

Pour être à nouveau confronté au public, Essl devient donc « *performeur* ». La performance est une pratique artistique présente dans toutes les disciplines de l'art contem-

porain, supposant des actions éphémères à caractère unique³⁵, dont par exemple John Cage ou Yoko Ono (1933-) font partie des initiateurs. La désignation est cependant exagérée : les « performances » d'Essl sont avant tout des improvisations³⁶. L'improvisation fait partie des terrains de jeu privilégiés de la création musicale contemporaine, comme en témoigne par exemple le compositeur américain George Lewis : « [L'improvisation] se porte bien un peu partout. Beaucoup de monde essaie d'apprendre à improviser, parce que cela a toujours été un moyen formidable pour découvrir comment faire de la musique. Il y a une ébullition très intéressante parmi les improvisateurs — de nouvelles manières de structurer, des idées élaborées pour intégrer des partitions à l'intérieur de l'improvisation, de nouveaux sons, un élargissement des notions d'*instrument*, de *virtuose*, et du rôle du performeur. »³⁷ [Appleton et al., 1986, p. 81].

La volonté d'Essl de jouer devant un public comme il l'avait fait avec les groupes de rock de sa jeunesse se heurte toutefois à un problème conséquent : il n'a aucun instrument dont il pourrait jouer. Bien qu'il ait comme nous l'avons vu joué du piano, de la guitare électrique et de la contrebasse, il n'a en 1997 plus aucune pratique instrumentale de haut niveau, et ne peut donc pas prétendre être ou devenir un instrumentiste de premier ordre. Là encore, il trouve la solution en capitalisant sur son savoir-faire informatique : « Il serait difficile d'imaginer mon travail sans les ordinateurs, même lorsque je fais de la musique instrumentale ; c'est grâce à l'ordinateur que j'ai pu quitter ma

35. Au sens large, la musique générée par les programmes de composition en temps réel aurait donc un aspect performatif, au même titre que tout art de l'improvisation ; ce goût pour l'éphémère fait sans doute partie des motivations d'Essl à se tourner vers cette pratique et à la désigner ainsi.

36. Au sens large donné par le chercheur — et performeur — américain Richard Schechner, toute improvisation est une performance : « la performance [est] l'événement, la manifestation artistique ostensible dans laquelle l'acte (ou le geste de l'exécution), quel qu'il soit, [...] a une valeur pour lui-même, donne lieu à une appréciation esthétique et axiologique distincte, est déterminé par une action spécifique et renvoie aussi bien à des pratiques multiples [...] qu'à des enjeux » [Schechner, 2008, avant-propos de C. Biet]. En se réclamant de ce terme, Essl évoque néanmoins dans le champ de la musique savante contemporaine des actions artistiques beaucoup moins traditionnelles qu'une improvisation, telles les performances de Cage.

37. « *I think it [improvisation] is a healthy situation overall. Many people are trying to learn to improvise, because it has traditionally been a wonderful way to learn about the possibilities of making music. There is a lot of interesting activity among improvisers — new methods of structuring, advanced ideas of how to integrate scores with improvisation, interesting new sounds, extended notions of what an instrument is, what a virtuoso is, what a performer's role is.* »

tour d'ivoire »³⁸ [Essl et Pagano, 2009]. Essl conçoit ainsi son propre « instrument », qu'il nomme *m@ze 2* (*Modular Algorithmic Zound Environment*, 1999-). Programmer son propre instrument est une pratique commune dans la musique contemporaine. Le compositeur américain Curtis Roads témoigne ainsi : « Beaucoup d'instruments digitaux peuvent être associés aux micro-ordinateurs de faible coût. La prolifération d'ordinateurs bon marché donne accès aux instruments intelligents à virtuellement n'importe quel musicien qui les souhaite. »³⁹ [Appleton et al., 1986, p. xviii].



FIGURE 7 – Karlheinz Essl réalisant une performance avec *m@ze 2*— Photo © Viktor Brázdil

Comme on peut le voir sur la photographie 7, *m@ze 2* combine à la fois du matériel de musique électronique (pédales, micros, console MIDI...) et une interface logicielle. La partie physique, tactile, permet à Essl de mettre au point une sorte de *geste instrumental*. Quant à l'interface logicielle, visible plus précisément sur la figure 8, elle incorpore des algorithmes de traitement du son et des *Strukturgeneratoren* pour la composition en temps réel, manipulant ainsi des données sonores aussi bien que symboliques. Le *m@ze 2* est donc un méta-instrument dans le sens employé ci-dessus. On peut même parler de « super-instrument », en ce qu'il regroupe de nombreux méta-instruments. Essl l'a en effet constamment développé et amélioré depuis sa création en 1998, y

38. « *Ohne Computer wäre meine Arbeit schwer vorstellbar, selbst wenn ich Instrumentalmusik mache; dadurch war es mir möglich, aus meinem kompositorischen Elfenbeinturm auszubrechen.* »

39. « *Many digital instruments can be attached to inexpensive personal computers. The proliferation of inexpensive computers puts the capability of intelligent instruments within the reach of virtually every musician who wants them.* »

incorporant la plupart de ses créations logicielles. Le mot « *Modular* » dans son nom reflète le fait que le *m@ze 2* ne possède pas de structure contraignante — mais donc pas non plus d'unité — comme un instrument au sens usuel (qu'il soit matériel ou logiciel). Il s'agit plutôt d'une collection de sous-programmes qui peuvent être exécutés séquentiellement ou en parallèle. Dans *Machine Musicianship*, le compositeur contemporain Robert Rowe identifie « trois classes d'algorithmes qui se prêtent particulièrement à l'interaction : le séquençage, la transformation et la génération. Le séquençage implique l'utilisation d'un matériau pré-enregistré qui est joué plus ou moins tel quel, mais où l'on contrôle les points d'entrée et de sortie, les boucles, la vitesse de lecture, etc. [...] Les techniques de transformation [...] prennent une source externe, lui appliquent une ou plusieurs transformations, et jouent ce matériau transformé en contrepoint de l'original. Les techniques de génération extraient un matériau de base d'un jeu de données fixe et lui applique des processus pour l'adapter, le prolonger ou l'embellir »⁴⁰ [Rowe, 2004, p. 203]. Ces trois classes sont présentes chez Essl. On voit en bas à droite de l'interface du logiciel (figure 8) que des extraits sonores pré-enregistrés sont présents, l'un d'entre eux étant entièrement représenté (en bleu), qui peuvent être séquencés grâce notamment au logiciel incorporé *REplay PLAYer* (2000-2001), programmé d'abord indépendamment par Essl. La transformation est illustrée par la figure 7, où l'on voit qu'Essl se sert du *m@ze 2* pour augmenter⁴¹ des instruments traditionnels. Enfin, les *Strukturgeneratoren* permettent la génération, dans le sens défini par Essl, plus large que celui de Rowe. Les modules de la *Lexikon-Sonate*, dont deux sont étudiés dans la partie 3.3.2, sont par exemple intégrés dans le *m@ze 2*. Cet instrument, qui rassemble la plupart des travaux du compositeur au cours de sa carrière, peut en somme être décrit comme une version condensée et ergonomique de sa table de travail, et destinée à l'improvisation.

40. « ... three classes of compositional algorithms that are particularly suited to interaction: sequencing, generation and transformation. Sequencing involves the use of pre-recorded material that is played back more or less as recorded, but with control over start and stop points, looping, playback tempo, and so on. [...] Transformations techniques [...] take input arriving from an external source, apply one or several transformations to it, and output the varied material as a counterpoint to the original. Generation techniques extract seed material from some stored repertoire and apply processes to adapt, extend, or embellish it. »

41. On parle en général d'*instrument augmenté* lorsqu'un musicien ajoute des fonctionnalités électroniques à son propre instrument. La guitare électrique est un instrument augmenté.

On peut noter qu'en utilisant des *Strukturgeneratoren* algorithmiques pour l'improvisation, Essl réalise là encore une synthèse dans une opposition classique, décrite par exemple dans *Composers and the Computer* : « [Tristan] Perich me dit que ses compositions émergent de l'improvisation, de l'esprit à l'œuvre — en général le piano, qui est son instrument de prédilection. Il oppose ceci à d'autres compositeurs qui utilisent des algorithmes, ce qui entraîne des complications. “Il y a une différence entre un processus qui fait partie de l'inspiration ou de votre ensemble d'outils, et un processus qui fait office de déterminant.” Il préfère le premier. »⁴² [Appleton et al., 1986, p. 264]. Il y a bien, dans l'utilisation du *m@ze* 2 pour l'improvisation, une composante « hors du temps », puisque les *Strukturgeneratoren* du programme ont été conçus en amont de la performance (il ne s'agit pas de *live coding*). Ce qui fait que la musique créée avec le *m@ze* 2 reste improvisée, c'est que les algorithmes ne sont utilisés que localement, à la demande et sous le contrôle du performeur, et non pour décider de la structure générale de la musique.

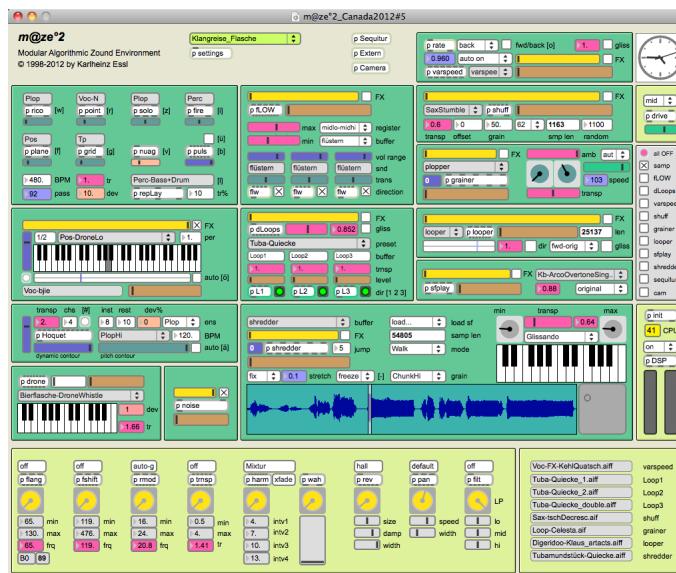


FIGURE 8 – L'interface graphique du logiciel *m@ze* 2— Photo © Karlheinz Essl

L'ouverture voulue par Essl passe aussi par la collaboration avec d'autres artistes, musiciens (instrumentistes et compositeurs) ou non. Cela l'amène à composer de nou-

42. « [Tristan] Perich tells me that his compositions spring from improvisation, the mind at play — usually at the piano, which is his main instrument. He contrasts this with other composers who use algorithms, which introduce complications. “There’s a difference between process being part of the inspiration or the tool set that you have, and process being a determinant.” He prefers the former. »

velles formes et pour de nouveaux cadres, comme l'écrit Julieanne Klein : « La production musicale d'Essl couvre tous les media possibles : orchestre, musique de chambre, théâtre musical, performance, musique électronique *live*, musique informatique et électronique, méta-compositions et compositions en temps réel, méta-instruments, installations et paysages sonores, musique de film, visuels, compositions textuelles et pièces pour instruments solo. Cherchant constamment à étendre sa production créative, Essl collabore fréquemment avec des artistes d'autres disciplines, y compris des chorégraphes, des danseurs, des plasticiens, des vidéastes, des architectes, des poètes, des écrivains, et des graffeurs. »⁴³ [Klein, 2013]. L'abolition des barrières entre disciplines artistiques, et en particulier la sonification des supports visuels, est une tendance importante du champ artistique à partir de la fin des années 1980. C'est l'avènement du *multimédia* (association de media différents, ici dans un même objet artistique), conséquence de la révolution numérique, et de l'*interartialité* (décloisonnement des champs disciplinaires dans l'art).

La concrétisation de cette nouvelle volonté d'ouverture passe par un projet d'envergure intitulé *fLOW*. Il s'agit d'une entreprise polymorphe autour d'un programme homonyme générant un paysage sonore ; elle donne lieu à dix installations et vingt-deux performances (principalement en 1999 et 2000, mais les dernières ont lieu en 2007). Toutes incorporent le *m@ze 2*, et chaque performance fait appel à des improvisateurs différents, jouant des instruments variés et issus de traditions musicales diverses (Nouvelle Musique, jazz expérimental, improvisation *live*, musique électronique).

Enfin, un autre événement important se produit en 1999 : l'ouverture du Essl Museum. Elle apporte deux explications supplémentaires à la nouvelle orientation de la carrière d'Essl. D'abord, elle s'inscrit dans la même ligne idéologique : Karlheinz Essl senior a fait construire ce musée pour mettre sa collection personnelle d'art contemporain à la disposition du public⁴⁴. Sur le plan matériel, elle offre de plus une plus grande

43. « *Essl's compositional output spans every possible medium : orchestral, chamber, musical theatre/performance, live electronics, electronic computer music, real-time and meta compositions, meta-instruments, installations and soundscapes, film music, visuals, text compositions and works for solo instruments. Always looking to expand his creative output, Essl frequently collaborates with artists from other fields, including choreographers, dancers, visual artists, video artists, architects, poets, authors, and graffiti artists.* »

44. La motivation de ce projet n'est naturellement probablement pas seulement philanthropique —

autonomie à Karlheinz Essl junior. Celui-ci dispose en effet d'un studio à l'intérieur du bâtiment, et d'un poste de conservateur chargé de la programmation musicale, grâce auquel il peut mettre ses productions en scène à l'intérieur des expositions, et nouer des relations en organisant des concerts.

Cette dernière période est donc celle d'un changement de la pratique d'Essl, plus que dans son esthétique ou dans ses méthodes de composition. À nouveau, ce tournant peut être mis en relation avec le contexte historique. Les progrès en informatique générale ont amené la musique numérique au grand public (les premiers baladeurs MP3 apparaissent ainsi en 1998), rendant obsolètes les travaux matériels en informatique musicale. Les institutions, comme l'Institut de Sonologie ou l'IRCAM, dont le principal avantage outre la transmission de compétences entre compositeurs et musicologues était de rassembler les moyens pour le développement de ces prototypes, perdent en rayonnement. Les progrès en informatique musicale ne sont désormais presque plus qu'exclusivement logiciels.

2.5 Conclusion

Essl apparaît comme emblématique d'une génération. Les précédentes avaient connu la naissance de l'ordinateur, les débuts de la science informatique et les promesses de la cybernétique, la suivante est née dans un monde déjà informatisé ; celle d'Essl a vu les ordinateurs quitter les institutions pour conquérir la sphère privée, et s'imposer dans de nombreux champs de la société. L'informatique représente ainsi à la fois une réalité concrète, et un nouvel horizon à explorer, dont les progrès tangibles se font à de courtes échelles de temps. Elle a accompagné Essl tout au long de sa carrière de compositeur, qui lui a accordé une place de plus en plus importante dans son travail. Véritable moteur de son évolution, elle a guidé ses choix et permis de les réaliser, ainsi que de développer son originalité.

sa vocation est en particulier sûrement aussi une entreprise de communication — ; elle n'est du moins certainement pas économique, la construction du musée par l'architecte Heinz Tesar et son entretien étant assurés par le patrimoine de la famille Essl, sans aide publique. La mise à mal de ce patrimoine par la crise financière de 2007 à même conduit à un changement d'actionnaire majoritaire et à la fermeture du musée en juin 2016.

3 Le rôle de l'informatique à travers ses programmes

3.1 Introduction

Cette partie se propose d'analyser la manière dont Essl a recours à l'ordinateur dans sa pratique artistique. Cette analyse partira du code même des programmes qu'il a conçus, selon l'approche du séminaire *Codes Sources*⁴⁵, ainsi décrite par Baptiste Mélès, chercheur en philosophie spécialisé dans l'informatique : « À celui seul qui prend la peine de les lire effectivement, les codes sources révèlent leur richesse. On y découvre que l'élégance d'un algorithme réside parfois hors de sa complexité, dans l'usage virtuose des idiomes du langage de programmation ou dans la connaissance fine de la machine à laquelle il est destiné. Bien souvent des codes sources comportent davantage de lignes de commentaires que de code. Tous ces trésors de pensée informatique fondent à la compilation comme neige au soleil. [...] Le but du séminaire est de décrire ces œuvres de l'esprit comme des textes à part entière. »

Nous commencerons donc par étudier les codes sources de deux créations majeures d'Essl : la bibliothèque *Realtime Composition Library*, collection de fonctions écrites dans le langage MAX/MSP qu'il utilise dans la plupart de ses compositions informatisées depuis 1992, et la *Lexikon-Sonate*, l'une des premières pièces écrites à partir de cette bibliothèque et le plus diffusé des programmes d'Essl. Cela conduira à discuter les notions, dont on verra qu'elles sont essentielles dans les travaux d'Essl, de « hasard » et surtout de « composition en temps réel ». Nous aborderons ensuite l'usage que le compositeur fait du réseau Internet. Pour finir, nous montrerons que les conclusions de ces différentes approches décrivent un rapport à l'ordinateur comme à un outil, purement technique.

3.2 La *Realtime Composition Library*

La *Realtime Composition Library* (*RTC-lib*) est une bibliothèque, c'est-à-dire un ensemble de fonctions réutilisables dans des programmes, développée par Karlheinz Essl dans le langage MAX/MSP. Son développement commence dès 1992, c'est-à-dire

45. Voir <http://codesource.hypotheses.org/> (consulté le 30 août 2016).

dès le moment où Essl a été mis en contact avec ce langage développé un an plus tôt par Miller Puckette⁴⁶, lors de son stage à l'IRCAM. Son origine remonte même à 1988, si l'on considère qu'Essl a commencé par implémenter dans ce nouveau langage les algorithmes de composition qu'il avait déjà développés en xLOGO, rassemblés dans la bibliothèque *COMPOSE*.

Cette bibliothèque est le socle de la plupart des œuvres réalisées par Essl impliquant la programmation. Il en recense 56 à ce jour : partitions, programmes, installations, *works-in-progress*..., parmi lesquels certaines de ses créations majeures comme les *Sequitur* (série de pièces pour divers instruments solo avec électronique *live*, 2008-2010), *fLOW* ou encore la *Lexikon-Sonate*, étudiée dans la partie 3.3. C'est aussi à partir d'elle qu'il a conçu l'« instrument » qu'il utilise lors de ses performances, le *m@ze* 2. Elle est distribuée sous licence libre : disponible en téléchargement, elle peut être utilisée et modifiée gratuitement, à la seule condition d'en citer les auteurs.

Dans ce qui suit, nous proposons, après une présentation générale du langage MAX/MSP, une description détaillée de cette bibliothèque, avec une classification de ses fonctions en deux catégories : les *fonctions ergonomiques* et les *fonctions compositionnelles*. On trouvera ensuite une analyse technique de ces fonctions mettant l'accent sur leur complexité, le type de pensée qu'elles révèlent ou encore la façon dont elles sont documentées, ce qui amène à dégager le projet esthétique qu'elles servent.

3.2.1 Description

MAX/MSP est un *langage de programmation graphique*, c'est-à-dire un langage de programmation dans lequel les programmes ne sont pas écrits en texte mais construits à partir d'éléments graphiques. Dans le cas de MAX, ces éléments sont des boîtes ou « *patches* » possédant des entrées et des sorties, et des liens qui permettent à ses fonctions d'interagir — typiquement : relier une sortie d'un *patch* à une entrée d'un autre. La figure 9 montre un exemple de programme simple. Ce type de représentation des programmes comme un assemblage de *patches* reliés entre eux est la plus commune parmi les langages de programmation graphique, et on la retrouve dans d'autres

46. Pour plus d'informations sur MAX, voir [Puckette, 1991]; une brève description du langage est faite ci-après.

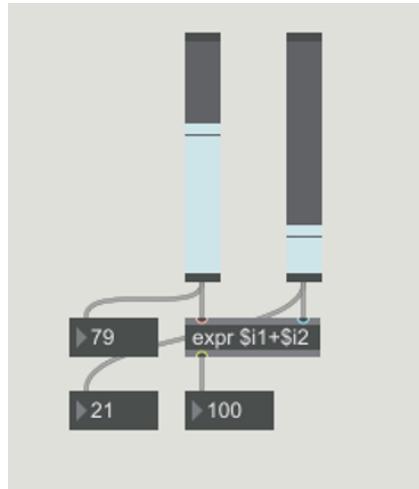


FIGURE 9 – Un programme basique en MAX/MSP : l'affichage et l'addition de deux entiers déterminés par des *sliders*.

environnements de CAO comme *OpenMusic*. Il existe deux modes d'édition distincts, l'un pour construire le « circuit » du programme, l'autre permettant de l'exécuter et de modifier les paramètres (valeurs numériques, impulsions ou « *bangs* », curseurs, etc.). MAX ayant été développé expressément pour la musique assistée par ordinateur, il contient de nombreux *objets* (fonctions) dédiés comme des entrées et sorties MIDI, ou consacrés au traitement du son⁴⁷.

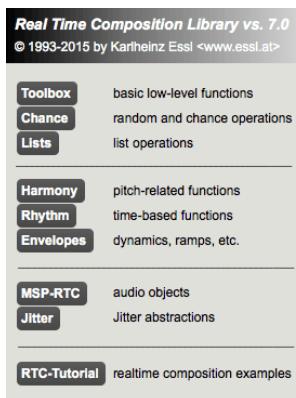


FIGURE 10 – Le menu et les huit rubriques de la *RTC-lib*. Photo © Karlheinz Essl

La *RTC-lib* compte à ce jour 177 objets en tout. 140 sont écrits par Essl, les autres en

47. En réalité, MAX ne permettait à l'origine que la manipulation de données au format MIDI ; c'est la bibliothèque MSP, ajoutée en 1997, qui permet le traitement de signal audio (*Digital Signal Processing*, ou DSP). Une autre bibliothèque d'importance nommée Jitter a été ajoutée en 2002, dédiée à la synthèse et à la manipulation graphiques.

collaboration avec ou d'après d'autres compositeurs ou chercheurs⁴⁸. Elle est divisée en huit rubriques, qu'il illustre en outre un tutoriel (voir figure 10). Les trois premières rubriques, *Toolbox* (outils divers), *Chance* (opération aléatoires) et *Lists* (traitement de listes), contiennent des objets « basiques ». Les trois suivantes, *Harmony* (gestion de hauteurs de notes), *Rhythm* (gestion du rythme) et *Envelopes* (gestion du volume), des objets « de composition », et les deux dernières, *MSP* et *Jitter*, regroupent les fonctions directement liées aux entrées et sorties son et vidéo. L'analyse qui suit propose un découpage différent. Les fonctions des rubriques *Toolbox*, *Lists*, *MSP* et *Jitter* seront appelées *fonctions ergonomiques*. Elles répondent en effet au besoin de simplifier la tâche du programmeur en rendant plus rapides⁴⁹ ou plus intuitives des procédures d'usage courant, qui pourraient aussi servir à la réalisation de programmes en-dehors de l'informatique musicale. Les autres rubriques regroupent les *fonctions compositionnelles*, c'est-à-dire les fonctions qui résolvent des problèmes spécifiquement artistiques et musicaux. Aux trois rubriques qu'Essl fait entrer dans cette catégorie s'ajoute donc la rubrique *Chance*.

Les fonctions ergonomiques

La plupart de ces fonctions sont très simples, en particulier dans la rubrique *Toolbox*. On y trouve ainsi des fonctions réalisant les calculs de l'inverse d'un nombre, de son opposé, un accumulateur (fonction qui fournit la somme de toutes les valeurs qu'elle a reçues au cours de l'exécution du programme, ou le nombre de fois qu'elle a été appelée)... D'autres corrigent ou améliorent des fonctions pré-existantes, comme la division euclidienne (qui ne fonctionne à l'origine que si le numérateur positif) ou l'arrondi (rendu paramétrable par Essl : au lieu de calculer seulement l'entier le plus proche de la valeur d'entrée, on peut arrondir à x près).

C'est aussi le cas de la rubrique *Lists*, dont la plupart des fonctions proposent simplement une forme plus ergonomique de fonctions pré-existantes. 14 d'entre elles se résument même à un renommage de fonctions contenues dans un objet nommé *z1*. Celui-ci fournit de nombreuses procédures de manipulation de listes, mais aux noms peu explicites. *Dans la RTC-lib*, *z1 rev* devient ainsi *reverse*, *z1 rot*, *rotate*, ou en-

48. R. Albert Falesch, Charles Baker, Frank Barknecht, John Chowing, Chris Dobrian, Richard Dudas, Gerhard Eckel, Peter Elsea, Philippe Gruchet, Gary Lee Nelson, Serge Lemouton, James McCartney, Iain Mott, Eric Singer, Les Stuck et David Zicarelli.

49. Rapidité au moment de la création du code, pas de son exécution.

core `z1 lookup` est légèrement modifiée pour devenir `nth`, qui renvoie le *n*-ième élément d'une liste fournie en entrée. La rubrique *Lists* contient ainsi principalement des opérations simples et classiques sur les listes : calcul de la longueur d'une liste, de l'intersection de deux listes, scission d'une liste... On en trouve d'autres qui résument en une seule fonction des opérations simples mais utiles et donc amenées à être utilisées souvent. Par exemple la fonction `butfirst` (respectivement `butlast`) cache simplement, en lui donnant une signification explicite, la scission d'une liste après son premier (respectivement avant son dernier) élément.

Enfin, les dernières fonctions essentiellement ergonomiques que contient la *RTC-lib* sont des conversions, présentes dans chaque rubrique. Elles permettent au compositeur de manipuler les différents outils de la bibliothèque sans se soucier de leur compatibilité de type, qui peut être immédiatement gérée par ces convertisseurs. Cela va de transformations de *pitch* (hauteur de note) en note au format MIDI ou en fréquence à celles entre différentes sorties sons, ou encore à celles entre durées relatives et absolues.

Les fonctions compositionnelles

Nous incluons la rubrique *Chance* dans cette catégorie, parce c'est à notre sens dans celle-ci que réside le cœur de la partie véritablement compositionnelle de la *RTC-lib*. En effet, les fonctions que nous avons qualifiées d'*ergonomiques*, bien qu'elles aient leurs spécificités et reflètent des partis-pris (comme l'importance accordée à la structure de liste), sont généralistes et répondent à des besoins d'abord *techniques*. La rubrique *Chance*, dédiée à la gestion de l'aléatoire⁵⁰, rassemble comme les précédentes des outils basiques et des fonctions simples utilisables dans d'autres contextes ; mais les utilisations possibles du hasard sont si nombreuses que les choix faits dans cette rubrique ont nécessairement une portée esthétique, sur laquelle s'appuient les modules *Harmony*, *Rhythm* et *Envelopes*.

Plus précisément, elle rassemble principalement des générateurs aléatoires dont la variété rend possible une grande liberté dans le développement d'algorithmes musicaux. Là encore, l'ergonomie est une préoccupation centrale. Tous les générateurs sont ainsi déclinés en deux fonctions, selon qu'ils fournissent leurs résultats séquen-

50. Ce que recouvre précisément la notion de « hasard » en informatique est discutée dans la partie 3.4.1.

tiellement, ou directement sous la forme d'une liste. Là encore, toutes les fonctions ont un code relativement simple, parfois largement fondé sur des fonctions standards de MAX, comme la génération sérielle, présente nativement dans la fonction `xrandom` et rebaptisée `series`.

Les générateurs aléatoires présents dans la *RTC-lib* implémentent des lois de probabilités classiques. On retrouve ainsi la probabilité uniforme (chaque événement possible a la même probabilité, comme sur un dé équilibré), sérialisme (la probabilité qu'un événement qui a déjà eu lieu se reproduise est nulle jusqu'à ce que tous les autres événements se soient produits chacun une fois), chaînes de Markov (déplacement sur un graphe donné, ou les probabilités de passage d'un état à un autre sont fixées à l'avance), mouvement brownien (aussi appelé « marche aléatoire », déplacement au hasard dans un espace donné tel que deux positions successives soient « proches »). Certains de ces générateurs sont déclinés en plusieurs variantes intégrant des paramètres supplémentaires comme des pondérations ou des interdictions de répétition. On remarque aussi la présence moins commune de plusieurs générateurs reposant sur des échelles logarithmiques, ainsi que l'absence de générateurs reposant sur des distributions de probabilité comme les lois de Gauss ou de Poisson. D'une manière générale, on trouve presque exclusivement — c'est-à-dire à l'exception du générateur de mouvement brownien, *brownian* — des générateurs *discrets* et non *continus*⁵¹. Les « principes de sélection » de Gottfried Michael Koenig, dont la *RTC-lib* et spécifiquement cette rubrique peuvent être considérées comme un développement, partagent avec elle cette particularité.

Dans les rubriques qu'Essl qualifie lui-même de compositionnelles, on peut distinguer trois sortes de fonctions : celles qui implémentent directement des méthodes classiques de composition, celles qui adaptent les générateurs aléatoires à cette fin, et les « super-fonctions ».

Les premières fonctions sont ainsi spécifiques à la composition et au son, elles implémentent des comportements, des outils, des gestes classiques. Par exemple, la fonction `neutral-harmony(n, i)` génère séquentiellement des notes à partir de n par

51. Des générateurs dont les productions appartiennent à des ensembles prédéterminés et explicitement finis, par exemple les entiers entre 1 et 12, et non (théoriquement, l'ordinateur ne pouvant en réalité manipuler qu'un nombre très grand mais fini de données) infinis, par exemple l'ensemble des nombres réels \mathbb{R} .

déplacement d'un intervalle i puis de son complémentaire ; la fonction `metro-dev%` modélise un instrumentiste humain en introduisant des approximations dans un battement parfaitement régulier ; la fonction `panning` calcule la distribution stéréophonique des volumes pour simuler le déplacement d'une source sonore. Parmi cette première sorte de fonctions, une sous-section entière de *Harmony* (16 fonctions) est dédiée aux manipulations dodécaphoniques usuelles (génération de séries, calcul de l'inverse ou de la rétrograde d'une série, etc.).

La deuxième sorte de fonctions est une application des différents générateurs aléatoires (et de certaines opérations de listes) à chaque rubrique. Par exemple, la fonction `brown-melody` génère séquentiellement des notes dont l'enchaînement suit un mouvement brownien, sans répétition d'octaves. Sur le plan purement technique, ce sont à peu de choses près des convertisseurs qui transforment les résultats de ces générateurs en données représentant des hauteurs de notes (entiers représentant des *pitches* ou *pitch-classes*⁵²), des rythmes (émission de *bangs*, ou entiers correspondant à des *entry delays* (ED)⁵³), et des volumes (entiers entre 0 et 127). Néanmoins cette description est réductrice, en ce que ces fonctions, considérées de l'extérieur, possèdent un véritable contenu sémantique et dépassent de ce fait la simple conversion de données.

Enfin, les « super-fonctions » combinent les précédentes. Elles n'apportent pas de possibilités créatives supplémentaires mais une interface pour choisir et alterner facilement entre les différents générateurs à disposition, en regroupant de plus de nombreux paramètres en entrée. C'est typiquement le cas du très complet `super-rythm`, visible sur la figure 11.

3.2.2 Analyse

L'analyse faite dans cette partie porte sur les aspects techniques et esthétiques de la *RTC-lib*. L'analyse technique étudie les choix directement liés à la programmation et

52. Le terme *pitch* renvoie à une hauteur de note au sens large, dans tout l'intervalle des fréquences audibles ; une *pitch-class* est l'une des douze hauteurs de la gamme. Un *pitch* correspond ainsi à l'association d'une *pitch-class* et d'un *registre*, qui indique dans quelle octave celle-ci se situe.

53. L'*entry delay* d'un événement est le moment de son exécution dans la pièce, ou plus généralement la durée entre un point de départ global (comme le début du morceau ou d'une section) et le début de cette exécution.

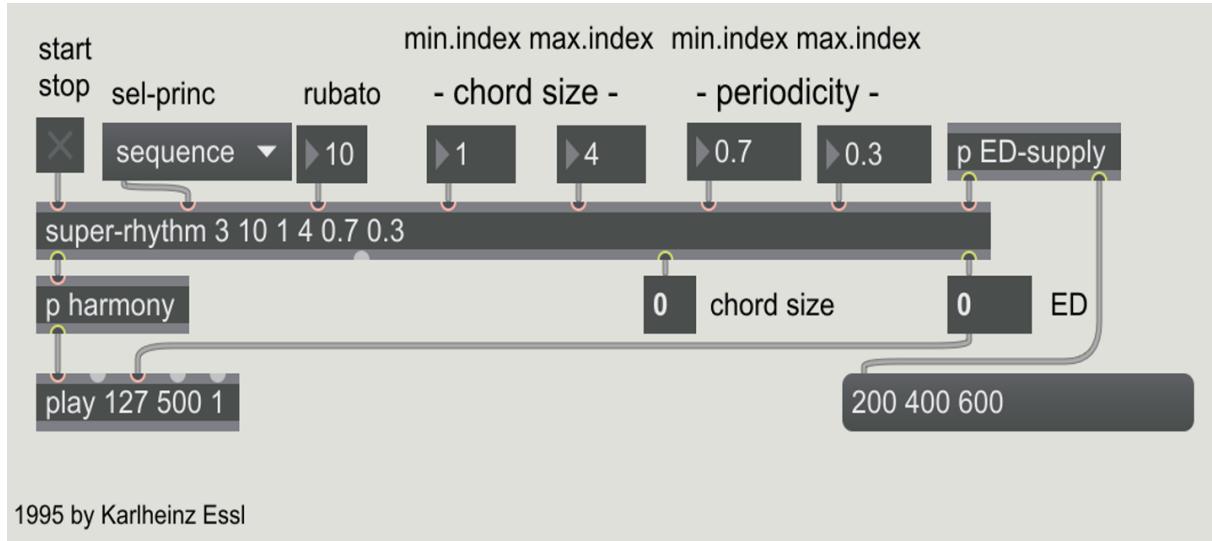


FIGURE 11 – La « super-fonction » *super-rhythm*, qui génère un rythme de manière largement paramétrable. Les ED générés sont sélectionnés parmi une liste fournie en entrée (*ED-supply*), selon un principe de sélection au choix (*sel-princ*, permet de sélectionner différents générateurs aléatoires). Une déviation aléatoire de ces valeurs d'origine peut être opérée (mesure par *rubato*). Enfin il est possible de régler la taille des accords (*chord size*, ou nombre de notes pouvant être jouées au même moment, donc au même ED) et la « périodicité » (*periodicity*), qui correspond à l'espérance mathématique du nombre de répétitions d'un ED immédiatement après avoir été sélectionné une première fois.

leurs conséquences, ainsi que le « style » de conception de code. L'analyse esthétique montre le projet et les parti-pris artistiques que révèlent et servent ces choix techniques. Rappelons que les informations obtenues par ces analyses disparaissent à la compilation, c'est-à-dire lorsque le code source est transformé en programme exécutable. Les choix d'implémentation faits par Essl n'ont pas d'influence sur le produit de ses algorithmes, et *a fortiori* sur la musique qu'ils génèrent ; ce sont sa pratique et sa pensée de programmeur qu'ils révèlent.

Analyse technique

Comme nous l'avons vu, la plupart des algorithmes présents dans la bibliothèque sont plutôt simples, mais il faut préciser ce que « simple » signifie. La manière théorique de quantifier la complexité d'un programme informatique est d'évaluer sa *complexité algorithmique*, qui correspond à l'ordre de grandeur du nombre maximal d'opérations élémentaires réalisées par une exécution en fonction de la taille de l'entrée⁵⁴. Dans le

54. Pour plus de détails sur cette notion centrale de l'informatique théorique, se reporter à [Knuth, 1998].

cas de la *RTC-lib*, tous les algorithmes ont une complexité *constante* ou *linéaire*⁵⁵, ce qui est effectivement faible. À titre de comparaison, de nombreux algorithmes courants comme le calcul du plus court chemin entre deux points d'un graphe ont une complexité algorithmique strictement supérieure. Les algorithmes de complexité constante s'exécutent en un nombre constant d'étapes, quelle que soit la taille de l'entrée (ou alors la taille des entrées est bornée). Les algorithmes de complexité linéaire nécessitent un nombre d'opérations élémentaires du même ordre de grandeur que la taille de l'entrée. En outre, les portions complexes (linéaires, donc) des fonctions de la *RTC-lib* ne sont jamais apparentes dans les *patches* établis par Essl ; si c'était le cas, elles apparaîtraient sous forme de boucles ou de récursivité (une fonction récursive peut s'appeler elle-même avant de terminer son calcul), totalement absentes : les calculs sont effectués de manière directe, sans retour en arrière⁵⁶. Ainsi les seules sources de complexité sont cachées dans les sous-programmes (des fonctions natives) utilisés à l'intérieur des fonctions de la *RTC-lib*, principalement les manipulations de listes et les objets de type `metronome`.

La pensée algorithmique que révèle la *RTC-lib* est avant tout fonctionnelle. Cela signifie que le paradigme avec lequel Essl approche la programmation lui fait voir les algorithmes comme des fonctions qui *calcurent* (pensée fonctionnelle) plutôt que par exemple comme des listes d'instructions à exécuter (pensée impérative). La fonction `remove` de la rubrique *Lists* illustre cette orientation : pour supprimer le n -ième élément d'une liste L , Essl calcule les sous-listes L_1 et L_2 séparées au niveau de cet élément, puis la liste L'_1 comme L_1 privée de son dernier élément, et enfin le résultat comme la jonction de L'_1 et L_2 . Une pensée impérative aurait plutôt fait reculer d'un rang chaque élément à partir du $n + 1$ -ième. Que la pensée soit fonctionnelle n'a rien de surprenant car c'est la manière de programmer naturellement induite par le lan-

55. Exception faite de la fonction `sort`, réécriture de la fonction `z1 sort`, qui trie une liste. Le tri est un problème algorithmique extrêmement classique, et les algorithmes qui le réalisent ont généralement une complexité de l'ordre de grandeur de $(n \cdot \log n)$ étapes pour une trier une liste de longueur n .

56. Les seules traces de récursion apparaissent lorsqu'il faut interdire certaines valeurs en sortie d'une fonction (répétition, octaves). Dans ce cas un élément fait office de « filtre », teste si le résultat est valide et, s'il ne l'est pas, lance une nouvelle exécution du reste de la fonction jusqu'à finir sur une valeur autorisée. Ceci ne change pas significativement la complexité des fonctions concernées car la probabilité que les filtres doivent demander une nouvelle exécution est faible.

gage MAX, et associée avec l'utilisation de listes, omniprésentes dans la *RTC-lib*⁵⁷. Il est cependant intéressant de noter qu'Essl, qui a commencé à programmer avec des langages impératifs (Basic, LOGO⁵⁸) maîtrise parfaitement les codes du paradigme de programmation le mieux adapté dans ce contexte — malgré l'absence de récursivité, autre caractéristique essentielle de la pensée fonctionnelle.

Par ailleurs, les algorithmes sont généralement codés de manière directe, concise, et très claire. On peut même remarquer parfois le choix, augmentant cette concision, d'étapes *hard-coded* : au lieu de faire calculer une sous-fonction de manière procédurale, l'ensemble de ses valeurs est intégralement décrit en donnée⁵⁹. Enfin, la documentation est très bien réalisée, avec des explications à la fois complètes et claires de chaque fonction, l'organisation en rubriques, et la présence d'exemples ainsi que d'un tutoriel. Cette documentation, nécessaire à l'entretien du code ou à son appropriation par d'autres utilisateurs, tient à la volonté d'Essl de mettre effectivement cette bibliothèque à l'usage de tous. La figure 12 présente le code de la fonction [neutral-harmony] de la *RTC-lib*, tel qu'Essl l'a conçu, en illustrant ces conclusions.

L'analyse de la *RTC-lib* montre ainsi qu'Essl programme d'une manière très bien adaptée au langage MAX. Il possède une maîtrise technique de son outil, adaptant ses compétences aux contraintes internes du langage de programmation qu'il utilise. L'efficacité de ses algorithmes et de la manière qu'il a de les présenter et de les documenter atteste en outre d'un savoir-faire plus général d'écriture et de diffusion de code informatique. Les fonctions de la *RTC-lib* restent cependant très simples ; leur intérêt pour Essl réside dans la teneur musicale de ce qu'elles permettent de produire, et elles n'en ont aucun en tant qu'accomplissements techniques.

Analyse esthétique

Il ne s'agit pas ici de faire une analyse esthétique approfondie, au sens courant, de la

57. La structure de données équivalente, en programmation impérative, étant le tableau.

58. LOGO permet la création de fonctions, implémente la récursivité, utilise des listes, et repose sur un langage fonctionnel, le LISP ; sa philosophie générale est cependant plutôt impérative.

59. Comme si par exemple une fonction calculant le carré de son entrée x , au lieu de calculer $x \times x$, rentrait la valeur de la x -ième case d'une liste fournie au départ de la forme $1, 4, 9, 16, 25, \dots$). Dans la *RTC-lib* ce *hard-coding* est bien sûr utilisé pour simplifier des problèmes plus élaborés que cet exemple, par exemple pour calculer la distribution panoramique des volumes en fonction de la position supposée de la source sonore.

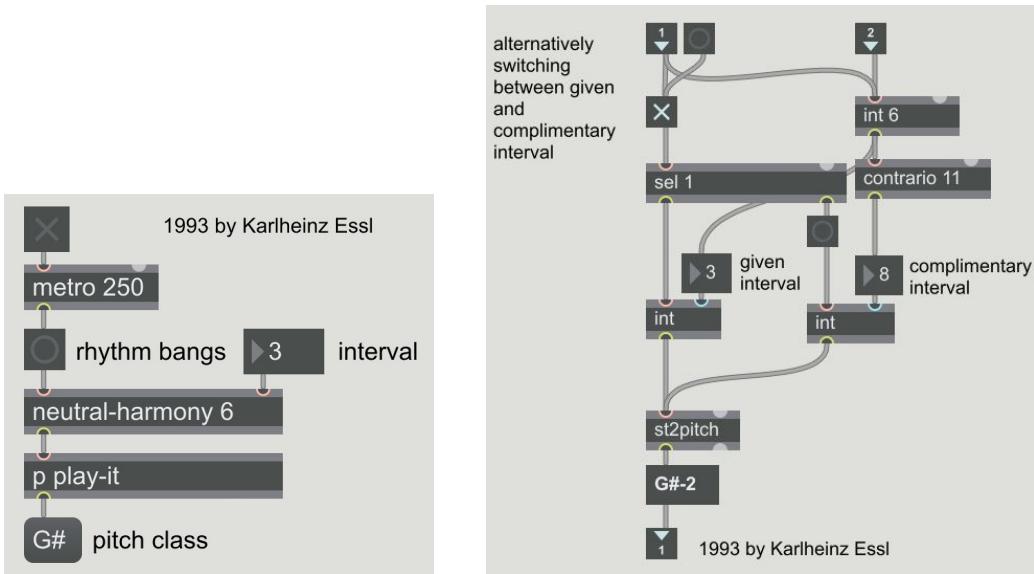


FIGURE 12 – À gauche, un exemple minimal d'utilisation de la fonction [*neutral-harmony*] de la *RTC-lib*, donnant à voir ses entrées et sorties, et à droite son code, programmé par Essl. Cette fonction produit, à chaque *bang* reçu dans son entrée gauche, des notes par déplacement d'un intervalle fourni par l'entrée droite et de son complémentaire. On voit qu'il s'agit d'une fonction peu complexe et que le code est documenté avec clarté : des commentaires indiquent les différentes étapes, et le calcul de l'intervalle complémentaire est réalisé par la fonction ergonomique [*contrario 11*], au nom explicite mais qui exécute simplement une différence entre 11 et son entrée. La pensée fonctionnelle d'Essl est aussi visible, par le choix de calculer à chaque fois et l'intervalle et son complémentaire, puis de choisir entre les deux (une fonction dans un style impératif aurait eu recours à une condition, sous la forme « *if... then... else* »). Enfin, la manière dont cette sélection est opérée révèle la maîtrise qu'a Essl de MAX. Elle est faite d'une façon concise et astucieuse grâce au *patch* en forme de croix, qui à chaque *bang* reçu change de statut (est activé ou désactivé), et au *patch* [*sel 1*] qui émet un *bang* dans sa sortie droite ou gauche selon que son entrée est active ou non. Ainsi, à chaque appel de la fonction principale, [*sel 1*] transmet un signal alternativement à droite ou à gauche, propageant l'intervalle d'entrée ou son complémentaire. Il s'agit d'une manière ingénieuse d'échafauder d'une façon courte et visuelle l'architecture générale de la fonction.

RTC-lib en tant qu'œuvre d'art contemporain, mais bien de mettre au jour les parti-pris de l'artiste Essl sous-jacents aux choix d'implémentation du programmeur Essl.

La première influence très nette est celle de l'école sérielle. Elle transparaît naturellement dans les fonctions directement dédiées aux calculs sériels (générateur aléatoire sériel, fonctions dodécaphoniques). Mais elle se ressent aussi à travers l'omniprésence des listes, structure de donnée naturellement adaptée à la représentation de séries, qui structurent l'ensemble des moyens de stockage de la bibliothèque. On peut en outre

relever l'influence directe de Gottfried Michael Koenig et de Karlheinz Stockhausen ; plusieurs algorithmes font des emprunts à leurs travaux et ils sont mentionnés dans les commentaires.

Par ailleurs, la représentation structurelle de la musique est très « pianistique ». Les notes sont en effet décrites par uniquement une hauteur, une durée et un volume, ce qui modélise bien un piano mais pas par exemple un violon, dont le son résultant d'un seul geste musical peut évoluer dans le temps. Il aurait pu en être autrement compte tenu de l'intérêt qu'Essl accorde au son en général, mais cela n'a rien d'étonnant dans un langage de programmation qui privilégie la notation MIDI, et un langage musical marqué par la pensée serielle.

Rythmiquement, on peut noter un affranchissement de la pulsation et de la notation sur partition, car ce ne sont jamais des noires, croches, doubles, etc., qui sont manipulées, mais exclusivement des durées absolues. Cette prise de liberté, qui s'associe à un rapport différent au temps de la musique est courante dans la musique du XX^e siècle ; le compositeur Guy Reibel écrit ainsi : « La pulsation disparaît au profit du voyage à l'intérieur de l'objet sonore, parcouru de son début à sa fin, lorsqu'il est formé, ou encore, plus radicalement, écouté hors du temps dans son écoulement et la variabilité de son flux lorsque sa durée est indéterminée [...] Le rapport au temps, à la notion de temps n'est plus le même, avec ces musiques qui "marchent toutes seules", dans lesquelles l'homme a délégué son geste (celui qui produit le son) à la machine. » (cité par [Heinrich, 2003, p. 89]). Enfin, lorsqu'il s'agit de générer les ensembles de valeurs parmi lesquelles opérer des choix aléatoires, Essl a une préférence nette pour les échelles logarithmiques, disponibles presque systématiquement. Ce choix traduit vraisemblablement la volonté de permettre à la fois précision et contrastes⁶⁰.

En conclusion, le projet esthétique qui s'appuie sur la *RTC-lib* peut être qualifié de combinatoire. La bibliothèque est conçue comme une *boîte à outils* plus que comme une collection d'algorithmes élaborés. Les fonctions qu'elle contient sont directes, et

60. Une échelle logarithmique comprendra par exemple de nombreuses valeurs basses et quelques valeurs élevées de l'intervalle qu'elle parcourt. Traduites en durées rythmiques, ces valeurs correspondent à de nombreux rythmes courts différents (d'où une grande précision possible), mais aussi quelques rythmes longs (d'où des contrastes possibles). Un générateur rythmique aléatoire opérant par sélection uniforme parmi une telle échelle logarithmique produira ainsi des rythmes globalement rapides et variés, ponctués de quelques ralentissements.

faciles d'utilisation ; elles sont techniquement simples mais très complètes, offrant une grande liberté de composition. La conception d'une œuvre avec la *RTC-lib* passe ainsi par la combinaison de ces fonctions (à l'instar des modules de la *Lexikon-Sonate*, étudiés dans la partie 3.3.2), et par le paramétrage des entrées qui peut engendrer des résultats musicaux très variés. Essl résume ainsi cette approche : « La plupart des objets [de la *RTC-lib*] sont orientés vers un traitement direct des données. L'utilisation de ces objets spécialisés rassemblés dans un *patch* clarifie et simplifie la programmation. Beaucoup de fonctions qui servent souvent en composition algorithmique sont disponibles dans cette bibliothèque — ainsi le compositeur peut se concentrer sur la composition et non sur les aspects de programmation. »⁶¹ [Essl, 2010].

3.3 La *Lexikon-Sonate*

La *Lexikon-Sonate* (commencée en 1992 et continuée jusqu'en 2007) est la pièce la plus connue de Karlheinz Essl. Elle naît de la rencontre de deux projets. D'une part, Essl vient de découvrir MAX et d'importer les algorithmes qu'il avait précédemment développés dans ce qui deviendra la *RTC-lib*. La *Lexikon-Sonate* lui permettra d'expérimenter l'application en temps réel de ces algorithmes dans une pièce complète. D'autre part, un groupe d'artistes viennois nommé *Libraries Of The Mind* conçoit un CD-ROM qui doit servir de support à un roman de l'auteur autrichien Andreas Okopenko (1930-2010), le *Lexikon-Roman* (1983). Cet ouvrage présente la particularité de narrer un récit fictif sous la forme d'un dictionnaire au travers duquel le lecteur est invité à voyager. L'idée du groupe *Libraries Of The Mind* est qu'il se prête particulièrement bien au format électronique, en tirant profit des liens hypertextes pour simplifier la lecture et les sauts d'une entrée à une autre. En outre, le projet propose d'enrichir le texte original en lui adjoignant des images et de la musique (faisant donc de ce CD un dispositif *multimédia*). Essl est alors invité à composer de courts extraits ou paysages sonores, chaque entrée du dictionnaire devant être accompagnée par un échantillon. Il refuse néanmoins cette approche, estimant que la musique accompagnant le *Lexikon-*

61. « *Most of these objects are geared towards straightforward processing of data. By using these specialized objects together in a patch, programming becomes much more clear and easy. Many functions that are often useful in algorithmic composition are provided with this library — therefore the composer could concentrate rather on the composition than the programming aspects.* »

Roman doit comme lui accorder une grande liberté au lecteur, par exemple changer nettement si celui-ci navigue rapidement d'une page à l'autre, ou au contraire conserver une ambiance stable si la lecture est calme.

C'est donc pour relever un double défi, technique et artistique, qu'Essl « compose », ou plutôt programme la *Lexikon-Sonate*. Dans ce qui suit, nous allons décrire la pièce et les différentes utilisations qui en ont été faites, puis proposer comme pour la *RTC-lib* une analyse d'une partie des algorithmes qu'elle emploie, et enfin dégager les conséquences que la forme de la *Lexikon-Sonate* entraînent pour l'œuvre et sa réception.

3.3.1 Description

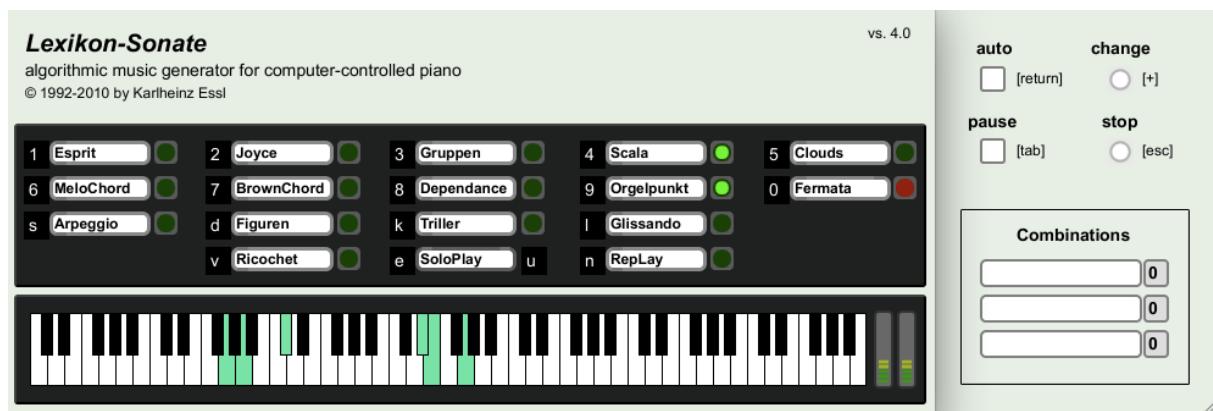


FIGURE 13 – Interface graphique de la *Lexikon-Sonate*. Photo © Karlheinz Essl.

La *Lexikon-Sonate* n'est pas un morceau au sens traditionnel du terme, mais un programme informatique qui génère de la musique. Reposant sur la *RTC-lib*, elle utilise en temps réel des algorithmes de composition algorithmique pour générer de la musique au format MIDI, qui est alors ou bien directement convertie en son de piano de synthèse par l'ordinateur sur lequel le programme est exécuté, ou bien jouée par un piano mécanisé comme le *Yamaha Disklavier*. Du fait de l'utilisation du hasard à plus ou moins grande échelle par les algorithmes implémentés en MAX par Essl, cette musique n'est ni pré-déterminée ni prévisible, et se renouvelle donc sans cesse. Le programme génère donc à chaque exécution un morceau qui pourrait être sans fin, et se crée au fur et à mesure qu'il est joué sans jamais se répéter.

Comme on peut le voir sur la figure 13, la *Lexikon-Sonate* utilise 17 « modules »⁶² (in-

62. Il en existe en réalité 24 en tout. Mais la figure montre la version libre de la *Lexikon-Sonate* ; les

titulés *ESPRIT*, *JOYCE*, etc.) équivalents. Chaque module est un algorithme, construit en combinant des fonctions de la *RTC-lib*. Ce sont ce qu’Essl nomme des *Strukturgeneratoren* : le produit de chaque module respecte une forme, une structure préétablie. Les dites structures recouvrent une large variété de fonctions et de complexités musicales. *GLISSANDI*, qui génère des *glissandi*, c’est-à-dire une série ascendante ou descendante de notes « consécutives » (séparées d’un ton ou d’un demi-ton)⁶³, est relativement simple à décrire, tandis qu’*ESPRIT* génère en théorie une mélodie *expressive* au sens de la musique classique viennoise. Les différentes structures peuvent selon les modules s’apparenter à des mélodies, des accompagnements, des « textures » ou des « figures » ; certains comme *FERMATA* sont même dédiés au silence, permettant à la musique composée de reprendre son souffle et de la segmenter, faisant entendre l’équivalent de parties et de sous-parties. Les programmes de deux de ces modules, *TRILLER* et *MELOCHORD*, sont étudiés en détail dans la partie suivante.

À chaque instant, exactement trois modules sont actifs. Ils génèrent chacun une séquence musicale correspondant à leur structure, de manière indépendante, qui sont jouées simultanément. Un changement des modules sélectionnés peut ensuite s’opérer (voir ci-dessous). La pièce jouée est ainsi la résultante du produit de trois *Strukturgeneratoren* totalement autonomes. Pour conférer une unité auditive à cette superposition, qu’elle ne soit justement pas entendue comme une confrontation absurde de sons mais paraisse véritablement téléologique dans sa construction, les trois modules actifs reçoivent par ailleurs une pondération (« premier plan », « second plan », « arrière-plan »), qui influence les paramètres réglant la gestion aléatoire du volume et de la proportion de silence. Essl affirme que les structures définies par les modules présentent des similarités avec des langages musicaux de l’histoire du piano, des *topoi* de cet instrument, « de Johann Sebastian Bach, Mozart, Beethoven, Schönberg,

24 modules ne sont disponibles que dans une version plus complète qu’Essl réserve pour ses propres performances et celles d’artistes qui lui en font la demande.

63. Il s’agit de la définition de la structure de *glissando* telle qu’elle apparaît dans la structure de ce module, et non de la définition générale, plus large et recouvrant des réalités différentes selon les instruments. Dans le cas de la *RTC-lib*, le format MIDI, essentiel dans MAX, impose au *glissando* d’être discret, comme s’il était exécuté sur un piano. À l’inverse, le *glissando* peut être envisagé comme *continuum* de fréquences entre deux notes, comme celui que jouerait un violoniste. Cette approche est par exemple centrale dans la musique de Iannis Xenakis, et par conséquent dans l’environnement de programmation *Iannis*, inspiré de ses travaux.

Webern, Boulez, [et] Stockhausen [à] Cecil Taylor ». Bien qu'il n'existe aucune citation directe des pièces pour clavier historiques, le programme est conçu de manière à ce que des formes musicales proches de celles-ci apparaissent au gré du hasard.

Il existe différentes manières de faire jouer le programme, autrement dit différents *modes de jeu*. À l'origine, la *Lexikon-Sonate* avait été imaginée comme une œuvre du type installation, avec une interactivité très limitée. La seule intervention possible de l'utilisateur était de provoquer le changement des modules actifs, qui s'opérait par remplacement du module présent depuis le plus longtemps, en suivant un principe sériel (après avoir été actif, un module donné ne sera plus réactivé tant que tous les autres modules n'auront pas été utilisés). Il s'agit d'une « super-série », puisque de nombreux modules utilisent aussi le principe sériel sur un niveau inférieur, et parce que son emploi pour combiner les modules n'a pas de conséquences facilement identifiables à l'écoute. Dans l'association originale avec le *Lexikon-Roman*, c'est cette commande de changement des modules actifs qui permet de calquer les variations de la musique sur le rythme de lecture. Un mode de jeu alternatif, sans interactivité, est également implémenté, dans lequel les remplacements de modules s'opèrent à des moments déterminés de manière aléatoire.

Plus tard, Essl élargit le champ des possibles du programme et en fait un métainstrument, en permettant à l'utilisateur de choisir lui-même quel module est activé et à quel moment. Cette version est distribuée librement en ligne ; dans une version plus élaborée réservée aux performances, le programme peut être relié à un contrôleur MIDI grâce auquel des paramètres globaux comme la « densité » du son (proportion statistique de moments de silence) peuvent être réglés par l'interprète-performeur.

Ces différents modes de jeu sont autant d'approches de la nature combinatoire⁶⁴ de l'œuvre. La grande variété du produit musical de la *Lexikon-Sonate* repose en effet à la fois sur la variabilité à l'intérieur de chaque module et sur les très nombreuses possibilités d'association de ces modules (2024 agencements de trois modules distincts sont possibles, 12144 en prenant en compte les trois plans sonores).

Le choix du piano pour cette pièce n'est pas anodin. Essl parle d'une revanche envers

64. Les propriétés combinatoires sont au cœur de nombreuses œuvres du XX^e siècle, par exemple en littérature les *Cent mille milliards de poèmes* (1961) de Raymond Queneau, recueil de cent quarante-vers pouvant être recombinés pour créer jusqu'à cent mille milliards de sonnets différents.

un instrument qu'il n'avait pas aimé apprendre lorsqu'il était enfant⁶⁵ ; c'est d'ailleurs un instrument pour lequel il a relativement peu écrit⁶⁶. On peut dégager trois motivations à ce choix. D'abord, le projet d'associer la pièce au *Lexikon-Roman* invite à adopter une démarche artistique proche de celle qu'il emploie. Or celui-ci est imaginé comme un pied de nez à la littérature classique, puisque le *Lexikon-Roman*, c'est-à-dire le roman-dictionnaire n'est pas un roman mais bien un dictionnaire, qui contient cependant des références aux formes classiques. Le choix du piano permet d'évoquer l'histoire et le répertoire extrêmement riches de cet instrument ; le titre, *Lexikon-Sonate*, ou sonate-dictionnaire, fait elle aussi allusion à la forme extrêmement classique de la sonate mais n'a évidemment aucun rapport avec elle dans sa construction. Elle ressemble de fait plus à un dictionnaire, avec cette banque de modules, qui permettent en outre à Essl de mettre l'idée de *Strukturgenerator* à l'épreuve au travers de cette recherche de *topoï* historiques. La deuxième raison est la justification du principe de superposition de trois modules simultanés. Comme le piano, ou plus généralement tout clavier, est l'instrument polyphonique traditionnel par excellence, sur lequel par exemple on « réduit » les œuvres orchestrales lorsqu'un instrumentiste seul veut en faire entendre la teneur musicale, il est plus aisément d'écouter la rencontre de structures potentiellement contradictoires lorsqu'elle s'opère avec le timbre de cet instrument. Enfin, le piano est aussi

65. « L'un des points de départ est ma relation entre amour et haine avec cet instrument, le piano. À sept ans, j'ai été obligé d'apprendre le piano ; je voulais jouer de la flûte à bec. Je ne suis cependant jamais devenu bon pianiste. C'était toujours très frustrant de voir que j'étais incapable de mettre adéquatement en œuvre sur les touches ce que j'avais dans la tête. Bien sûr, mes premières recherches de compositions furent cependant des pièces pour piano, mais depuis je n'ai écrit aucun autre morceau pour le piano, et j'envisage difficilement d'en écrire à nouveau. Tout ce que je pouvais faire pour le piano, c'était une pièce qui obéisse à des critères absurdes. » [Ehrler, 1999] (« *Einer der Ausgangspunkt ist meine Hassliebe zu diesem Instrument, dem Klavier. Ich wurde mit sieben Jahren gezwungen, Klavier zu lernen; ich wollte Blockflöte spielen. Trotzdem war ich am Klavier nie gut. Es war immer frustrierend, zu sehen, dass ich das, was ich im Kopf hatte, auf den Tasten nicht adäquat umsetzen konnte. Meine ersten Kompositionversuche waren natürlich trotzdem Klavierstücke, aber ich habe sonst kein einziges Klavierstück geschrieben bis jetzt, und ich kann mir auch nicht vorstellen, ein Klavierstück zu schreiben. Das einzige, was mir möglich war: Ein Klavierstück zu machen, das ganz absurde Kriterien erfüllt.* »).

66. Outre la *Lexikon-Sonate*, on recense seulement cinq pièces pour piano : *Con una certa espressione parlante* (1985), *Take the C Train* (2009), *juncTions* (2011/2012), *Gold.Berg.Werk* (2010-2014) et *STERN* (2013). Et *Take the C Train* est la seule autre œuvre pour piano solo (dans la mesure où l'on peut désigner ainsi la *Lexikon-Sonate*).

l'instrument dont les modèles automatisés comme le *Yamaha Disklavier* sont le plus répandus. Il s'agit par conséquent d'un choix naturel dans la conception d'une œuvre sous forme de programme contrôlant un instrument.

3.3.2 Étude de deux modules : TRILLER et MELOCHORD

Cette partie propose, comme ci-dessus pour la *Realtime Composition Library*, une étude détaillée du programme de la *Lexikon-Sonate*. Plus précisément, c'est TRILLER, l'un des « modules » ou *Strukturgeneratoren* qui composent la pièce, qui fait l'objet d'une analyse complète. Un autre module intitulé MELOCHORD est ensuite décrit de manière moins approfondie mais de façon à dégager les points communs avec TRILLER.

Module TRILLER

Le module TRILLER est un générateur de *trilles*. Traditionnellement, un trille est un ornement musical réalisé en alternant rapidement deux notes consécutives (séparées d'un ton ou d'un demi-ton). Cette définition est ici élargie : il s'agit d'une alternance rapide des notes d'un ensemble de 2 à 7 éléments, couvrant au plus une octave.

La figure 14 montre les deux parties principales du générateur de trilles⁶⁷. Il faut d'abord déterminer aléatoirement les différents paramètres du trille (quelles notes, à quel volume, etc.), et ensuite celui-ci peut être généré en temps réel. Il y a ainsi en quelque sorte une étape « hors du temps », la détermination des paramètres globaux, et une partie véritablement au fil du « temps réel », la génération de chaque note.

Comme on le voit sur la deuxième partie de la figure 14, la génération des notes du trille est commandée en premier lieu par le rythme. La fonction [p rhythm] émet un signal à chaque moment où une note doit être jouée, qui active les fonctions [p notes],

67. Les figures sont tirées des fonctions programmées par Essl en MAX/MSP. Précisons pour le lecteur qui souhaiterait les comprendre plus en détail que certaines liaisons sont masquées pour plus de visibilité. En outre, les *patches* dont le nom commence par la lettre *s* (*send*) ou *r* (*receive*) font de même office de liaison : la valeur reçue par [*s label*] est transmise à [*r label*]. Enfin, les noms de *patches* commençant pas *p* sont locaux : ceux-ci masquent un circuit complexe, mais le même nom utilisé à l'intérieur d'une fonction différente ne correspondra pas au même circuit. Par exemple, les deux *patches* nommés [*p rhythm*] dans les deux images de la figure 14 correspondent à des circuits différents. Ils sont seulement nommés ainsi pour indiquer que les circuits qu'ils représentent traitent tous deux d'aspects rythmiques.

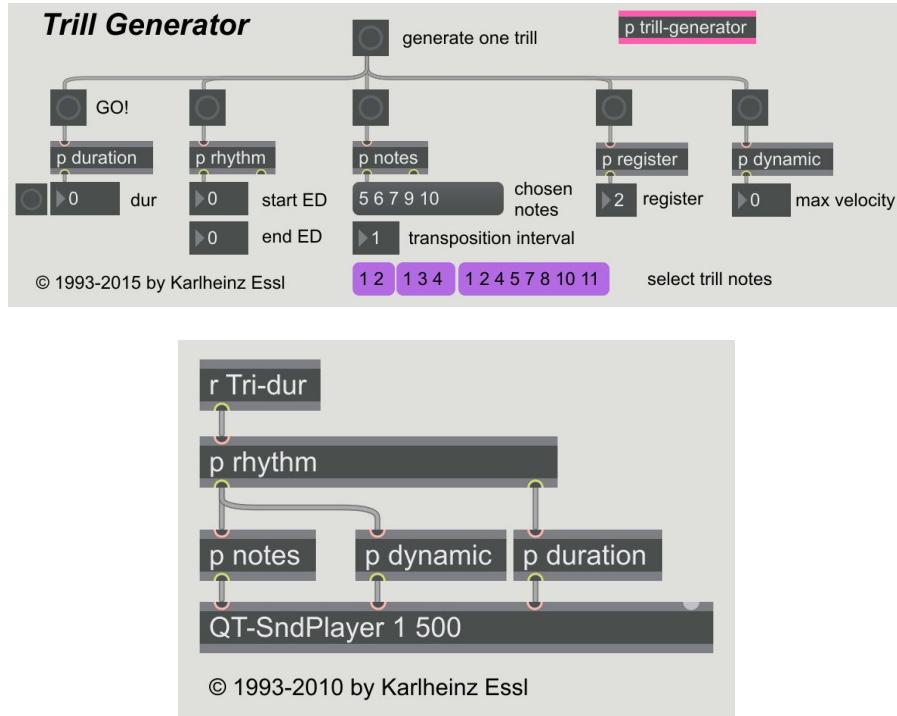


FIGURE 14 – Les deux parties principales du programme du module TRILLER. L'image du bas correspond à la génération des notes du trille en temps réel ; celle du haut regroupe les paramètres qui commandent celle-ci. Le circuit du haut est masqué par le *patch* rose [*p trill-generator*] de celui du bas.

[*p dynamic*] et [*p duration*], lesquelles déterminent chacune un paramètre (respectivement la hauteur, le volume et la durée) de cette note. Le synthétiseur [QT-SndPlayer] reçoit ces informations et exécute la note correspondante. La subordination au rythme des autres paramètres du son dans l'ordre de détermination a d'importantes conséquences sur le processus de composition. Il est en effet plus compliqué ainsi de construire une ligne mélodique et de lui attribuer ensuite un rythme, par exemple de ralentir à la fin⁶⁸. Cette remarque montre que, bien que permettant en théorie de créer toutes sortes d'algorithmes (ne seraient-ce que tous ceux réalisables par un compositeur humain, sans ordinateur), les environnements de CAO sont des outils comme les autres, dont les contraintes orientent les parti-pris de leur utilisateur.

En tout, sept paramètres globaux régissent l'ensemble du trille produit, visibles en bas du premier circuit de la figure 14 : *dur*, *start ED*, *end ED*, etc. Ici, ils sont déterminés

68. Cela est plus compliqué ou moins évident, mais reste possible. Il faudrait placer le générateur mélodique en amont, qui fournit son produit à la fois à une liste et au générateur de rythme. Les signaux émis par celui-ci commanderont alors l'envoi progressif des notes de la liste au synthétiseur.

lorsque le bouton central est activé, et dans la *Lexikon-Sonate*, à chaque fois que le module est appelé. La durée totale du trille (*dur*), les durées minimale et maximale entre deux notes successives (*start ED* et *end ED*, décidées par la fonction visible à gauche de la figure 15), le volume maximum (*max velocity*), l'intervalle de transposition (*transposition interval*) et le registre (*register*) suivent une loi de probabilité uniforme parmi l'ensemble de leurs valeurs possibles ; c'est la fonction [*between*] de la *RTC-lib* qui opère la sélection. Pour les trois premières, comme on peut le voir sur la figure 15, les valeurs possibles sont réparties sur une échelle logarithmique entre les deux valeurs extrêmes, calculée par la fonction [*trans-log*]. La détermination du matériel harmonique est plus élaborée, présentée à droite de la même figure. L'ensemble des notes de la gamme chromatique, représentées par les entiers de 1 à 12, est d'abord mélangé aléatoirement par la fonction [*scramble*]. La fonction [*between*] sélectionne uniformément le nombre de notes à conserver entre 2 et 6, et le transmet à la fonction [*slice*] qui scinde adéquatement la liste mélangée. Le résultat est enfin trié par [*sort*]. Cet algorithme permet effectivement de construire un ensemble aléatoire avec une probabilité uniforme (tous les produits possibles ont une probabilité identique). D'autres manières de parvenir au même résultat, y compris avec des fonctions de la *RTC-lib*, étaient possibles, par exemple en choisissant des notes parmi les douze possibles selon le principe sériel. La méthode employée par Essl témoigne à nouveau d'une pensée fonctionnelle plutôt qu'impérative.

Ces paramètres sont fixés pour tout le trille ; c'est à partir d'eux qu'est calculée chaque note. Certains des dits calculs utilisent des générateurs aléatoires. Le hasard intervient ainsi à deux échelles différentes dans le programme. Le rythme suit, à nouveau, une échelle logarithmique, ce dont il résulte un effet global de ralentissement. Cette échelle est calculée par la fonction [*ED-trans*] de la *RTC-lib*, afin que les durées entre chaque note varient de *start ED* à *end ED*, sur une durée totale de *dur* millisecondes. À chaque signal rythmique, une note est générée. L'une des hauteurs pré-sélectionnées est choisie au hasard avec interdiction de répétition par la fonction [*permute*]. Elle est ensuite « additionnée » aux deux paramètres *transposition interval* et *register* qui définissent la hauteur absolue de la note la plus basse du trille, et permettent donc à celui-ci d'être exécuté sur potentiellement n'importe quelle partie du clavier. Le calcul du volume sur l'ensemble du trille utilise la seule fonction [*schweller*], qui réalise un

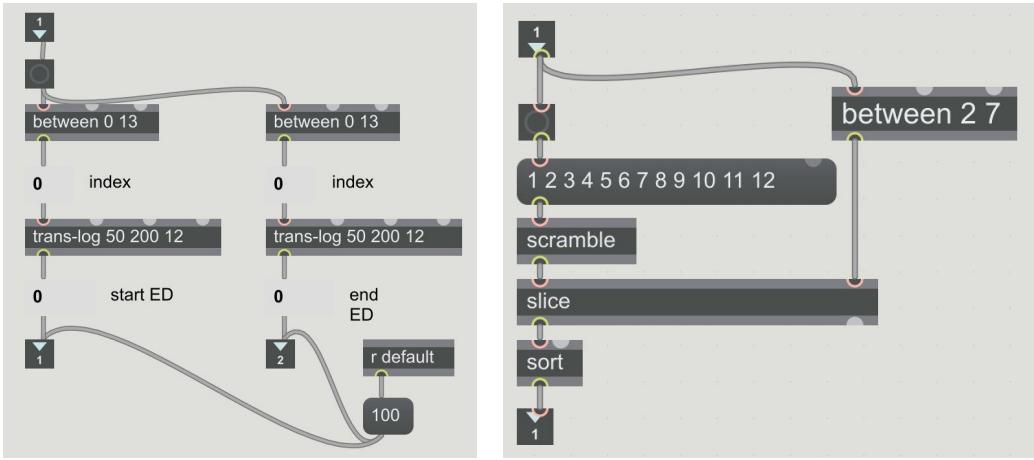


FIGURE 15 – Deux sous-circuits du module TRILLER utilisés pour la génération aléatoire des paramètres globaux. À gauche, on voit que les *entry delays* (ED) minimal et maximal sont sélectionnés à l'intérieur d'une échelle logarithmique de 12 pas entre 50 et 200 millisecondes. À droite, c'est la sélection des notes du trille qui est opérée : mélange des entiers de 1 à 12 (qui représentent les douze notes de la gamme chromatique), récupération des 2 à 6 premiers, et enfin tri.

crescendo suivi d'un *decrescendo* entre le silence complet et le paramètre *max velocity*. La durée de chaque note est aussi établie de manière déterministe, comme une fois et demie la durée qui la sépare de la suivante, ce qui donne un effet « lié » au résultat entendu.

Module MELOCHORD

Le module MELOCHORD génère des mélodies dont les notes peuvent éventuellement être enrichies par un accord. La figure 16 montre que le principe d'organisation est identique à celui rencontré dans TRILLER. Un premier circuit génère les paramètres globaux de la mélodie, et un second crée celle-ci note par note, sous l'impulsion d'un générateur rythmique.

Ce module est visiblement plus complexe que le précédent, régi par douze paramètres différents. Il s'agit cependant d'une complexité *horizontale*. Chaque génération de note implique plus de valeurs en entrée mais la profondeur du circuit est du même ordre de grandeur, et dans les deux cas il n'y a pas de boucle, comme dans la *RTC-lib*.

À nouveau, les paramètres généraux sont majoritairement sélectionnés à partir de distributions uniformes, souvent parmi des échelles logarithmiques. Les hauteurs de notes utilisent des intervalles calculés par l'une des fonctions purement musicales de la *RTC-lib*, [choose intervals], qui garantit des propriétés sonores du matériau généré

(la somme de deux intervalles ne peut être une octave, on ne peut pas construire d'accord diminué d'après ces intervalles, etc.). Pour les paramètres de distances entre les notes (ED), de registre, et de durée (de chaque note cette fois, et non plus de la structure totale), on remarque la présence d'une option nommée « périodicité ». Elle correspond à la probabilité pour chaque nouvelle note de conserver les valeurs de la précédente plutôt que de faire un nouvel appel au générateur aléatoire, afin d'obtenir par moments des unités sonores. Sur la figure 16, la « périodicité » est élevée pour les notes courtes (0.91) et nulle pour les notes longues. Ainsi, lorsqu'une note courte est créée, il est probable qu'elle commence une phrase enchaînant des notes de même durée. D'une manière générale, le produit de ce module devant être plus varié que celui de TRILLER, il fait appel à plus de fonctions complexes de la *RTC-lib*. La génération du volume fait ainsi appel à un mouvement brownien. Quant au générateur de rythme, il utilise la fonction [repchord-rhythm], qui dépend de six paramètres : les valeurs minimales et maximales d'ED, le nombre de pas pour l'échelle logarithmique calculée entre ces extrêmes, deux valeurs de « périodicité » qui varie sur cette échelle, et la taille maximale des accords générés (autrement dit le nombre de notes qui peuvent être générées simultanément).

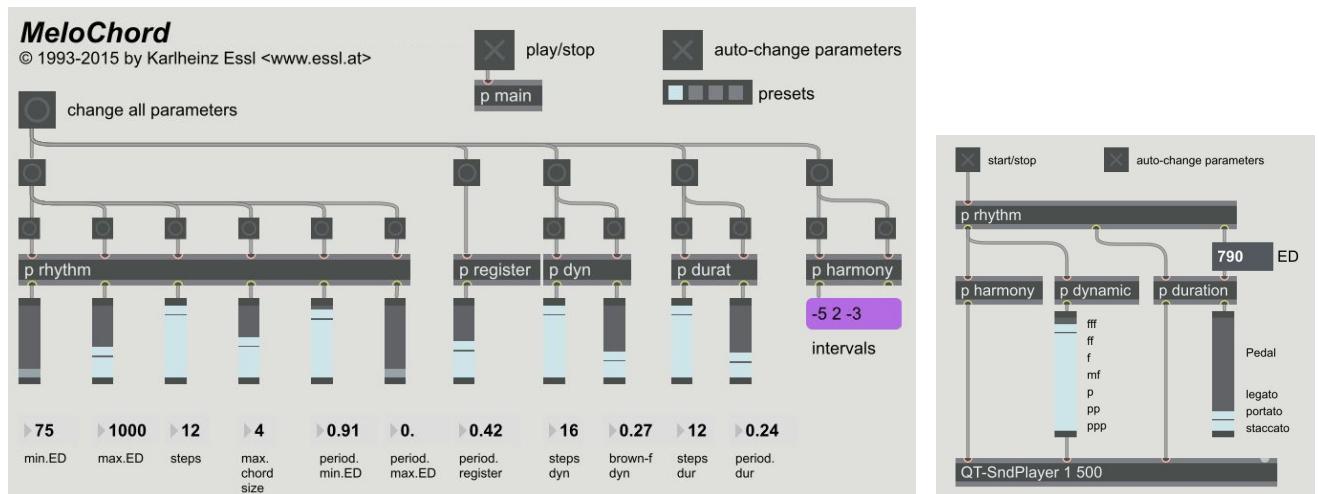


FIGURE 16 – Les deux parties principales du programme du module MELOCHORD. L'image de droite correspond à la génération des notes en temps réel ; celle de gauche regroupe les paramètres qui commandent celle-ci. Le circuit de droite est masqué par le *patch* [p main] de celui de gauche.

Conclusion

Les principaux enseignements que l'on tire de l'étude de ces deux modules sont leurs

similitudes dans la manière de programmer avec les fonctions de la *RTC-lib*, la façon dont cette bibliothèque est utilisée selon une méthode généralisable, et l'importance accordée au hasard dans le langage musical.

Ainsi les conclusions formulées sur la *RTC-lib* peuvent aussi s'appliquer à ces deux programmes de niveau supérieur. Les fonctions sont codées de manière directe, sans boucles sources de complexité, et commentées avec clarté. La pensée est principalement fonctionnelle, et traite le matériau musical avec des structures de listes. Bien que les deux modules soient relativement simples dans l'architecture de leur code, ils sont fortement paramétrables, ce qui permet de générer des résultats musicaux complexes et variés. La stratégie de genèse de ces modules n'est pas de suivre la logique du langage pour concevoir des structures mathématiques mais de faire obéir le code à des objectifs musicaux. On a de plus vu apparaître les contraintes inhérentes à l'environnement MAX/MSP qui ont incité Essl à subordonner au rythme les autres paramètres musicaux, montrant que, malgré les nombreuses nouvelles perspectives qu'elle ouvre et sa vocation universelle, la programmation possède comme tout outil technique des contraintes internes.

L'utilisation des fonctions de la *RTC-lib* est effectivement un projet d'ordre combinatoire, et les générateurs aléatoires de la rubrique *Chance* de la bibliothèque jouent un rôle fondamental. Chaque module repose sur un agencement de ces fonctions, dont l'intérêt ergonomique est exploité. Ils permettent en effet de donner un aspect intelligible à ces programmes, qui combinent les fonctions de la *RTC-lib* comme celles-ci les fonctions élémentaires du langage. On remarque aussi que dans ces deux modules, les fonctions compositionnelles les plus complexes (en terme de variété des résultats) sont rares, mais que le processus de génération est organisé autour d'elles. Les fonctions plus simples comme le générateur uniforme [*between*], présent à de multiples reprises, les complètent, avec un impact moindre sur la complexité sonore de la musique créée.

Enfin, l'omniprésence du hasard montre que l'élément aléatoire est un lieu commun (un *topos*) du langage musical d'Essl. Ce n'est plus une curiosité comme dans les jeux de dés musicaux, ni un enjeu comme dans la musique de Cage ou de Xenakis, mais un accessoire supplémentaire parmi les outils musicaux du compositeur. Le hasard est ainsi présent sur deux niveaux dans chaque module : d'abord pour sélectionner les

paramètres globaux, ensuite pour la génération de chaque note. Une fois que les dits paramètres globaux sont fixés, le module répond à la définition de *Strukturgenerator* puisque son produit respecte une certaine structure mais n'est pas prévisible ; pour le module complet, on pourrait donc même parler de *Strukturgenerator* généralisé.

3.3.3 Analyse

À nouveau, l'objectif de cette partie n'est pas de proposer une analyse esthétique approfondie de la pièce, mais de déterminer les spécificités d'une telle œuvre, en particulier dues à l'utilisation, d'une manière inédite lors de la création de la *Lexikon-Sonate*, de l'outil informatique.

Essl explique⁶⁹ que la *Lexikon-Sonate* est de par sa forme une sorte de métaphore de la vie humaine. Comme elle, elle est en perpétuelle évolution. Comme dans la vie humaine, des échos du passé y font régulièrement surface. Et de la même manière que l'être humain donne spontanément du sens aux nombreux événements et coïncidences qui se produisent en permanence, le principe de la pièce repose sur trois voix indépendantes mais dont la superposition doit faire croire à un réseau de sens⁷⁰.

La première caractéristique évidente de la *Lexikon-Sonate* est qu'il s'agit d'une œuvre ouverte. Cette notion assez large qualifie une œuvre dont certaines facettes ne sont volontairement pas maîtrisées ou arrêtées par l'auteur, mais laissées à celui qui la reçoit ou à des agents extérieurs — en particulier, il s'agit généralement d'œuvres qui abandonnent l'idée d'un « message » à transmettre au profit d'un champ d'interpré-

69. Voir entretien en annexe.

70. Naturellement, une telle métaphore est elle-même signifiante de la représentation du monde d'Essl. Il n'est pas anodin que cette *Weltanschauung* à une époque où la science est dominée par la physique quantique, les lois du chaos et les systèmes complexes, soit gouvernée par des processus aléatoires, alors qu'à la période classique, où le monde était décrit comme régi par un Dieu omniscient et omnipotent, la musique obéissait à des contraintes formelles et thématiques très strictes. Ce parallèle n'est toutefois qu'une ébauche dans la comparaison des représentations du monde historiques apportées par la science et l'art. On trouvera par exemple des remarques plus approfondies sur l'influence de la description scientifique du monde sur le champ artistique dans la première partie de [Eco, 1962]. Eco écrit par exemple : « La propagation de certaines notions dans son milieu culturel [influence l'] artiste, au point que son art veut et doit être considéré comme la réponse de l'imagination à la vision du monde répandue par la science : *l'art est une métaphore structurale de cette vision* » (p. 21).

tations possibles. Le concept est développé par Umberto Eco dans *L'Œuvre ouverte*. Eco y fait remarquer que la définition peut être appliquée à n'importe quelle œuvre d'art, puisque la représentation mentale qu'elle induit chez celui qui la consomme varie d'un individu à l'autre, étant nécessairement soumise à l'interprétation. Mais nombre de productions artistiques du XX^e siècle se caractérisent par la mise en exergue de cette aspect intrinsèque de l'œuvre d'art. La pièce "4'33'" (1952) de John Cage est ainsi un exemple minimal et éloquent d'œuvre ouverte : elle consiste en quatre minutes et trente-trois secondes de silence de la part de l'interprète, tout en affirmant que chaque son produit dans la salle de concert pendant l'exécution de la pièce fait partie de celle-ci ; l'auteur est alors entièrement dépossédé de son œuvre qu'il soumet à la contingence des conditions de sa réalisation.

Pour montrer que la *Lexikon-Sonate* s'inscrit effectivement dans une problématique d'œuvre ouverte, nous allons mettre en avant trois caractéristiques de cette pièce qui participent de cette problématique : l'indétermination, la mise en retrait de l'auteur, et la remise en question de la notion même d'œuvre d'art.

L'indétermination, la place laissée à la contingence, est essentielle dans l'idée d'œuvre ouverte. Elle s'oppose en cela aux œuvres « fermées » produites par les siècles précédents, c'est-à-dire les œuvres univoques, que critique par exemple Wittgenstein : « Il me vient à l'esprit que dans les discours sur les objets esthétiques, on emploie des expressions telles que : “Il faut voir cela ainsi, c'est pensé ainsi”, [...] “Tu dois entendre cette mesure comme une introduction”, [etc.] (cela vaut aussi bien à propos de l'écoute que du jeu). »⁷¹ [Wittgenstein, 1953, II, xi]. Eco décrit la réponse suivante des artistes à cette critique : « Aujourd'hui, l'accent est mis sur le processus, sur la possibilité de saisir *plusieurs ordres*. Dans la réception d'un message structuré de façon ouverte, l'*attente* implique moins une *prévision de l'attendu* qu'une *attente de l'imprévu*. » [Eco, 1962, p. 105]. Il est clair que l'omniprésence de l'élément aléatoire, rendue possible seulement par l'ordinateur et une pièce sous forme de programme informatique, crée cet imprévu en permanence.

La mise en retrait de l'auteur, de l'artiste, qui renonce en quelque sorte à sa position de

71. « *Da fällt mir ein, daß in Gesprächen über ästhetische Gegenstände die Worte gebraucht werden: „Du mußt es so sehen, so ist es gemeint“; „Wenn du es so siehst, siehst Du, wo der Fehler liegt“; „Du mußt diese Takte als Einleitung hören“; „Du mußt nach dieser Tonart hinhören“; „Du mußt es so phrasieren“ (und das kann sich auf's Hören wie auf's Spielen beziehen)* »

démiurge pour partager la paternité de sa production (typiquement avec le récepteur de l'œuvre, soit par l'interactivité, soit par l'interprétation dans une perspective proche du constructivisme radical cher à Essl), est un autre aspect important de l'œuvre ouverte. On pourrait la confondre avec l'indétermination ou du moins penser qu'elle est impliquée par elle, mais il n'en est rien. Si Essl avait construit sa pièce comme un unique *Strukturgenerator*, sa position de créateur aurait été plus affirmée : certes le hasard déterminerait certaines modalités, mais l'auteur serait en mesure de décrire la structure de n'importe quelle exécution. Dans la *Lexikon-Sonate*, le retrait est véritablement opéré par l'intrication de trois modules indépendants. À travers ce choix, Essl accepte l'impossibilité de prévoir même l'effet général produit par la pièce, et d'être lui-même surpris. Cette caractéristique, à la différence des deux autres étudiées ici, relève d'un parti-pris artistique et non de l'exploitation des nouvelles voies ouvertes par l'outil informatique.

Enfin, les propriétés de l'œuvre ouverte font qu'elle interroge la définition même de ce qu'est une œuvre. Adorno écrit ainsi : « Contrainte par la logique de ses propres faits, la musique, en un mouvement critique, a dissous l'idée d'œuvre achevée et rompu avec le public. [...] Les seules œuvres qui comptent aujourd'hui sont celles qui ne sont plus des "œuvres". » [Adorno, 1949, p. 41-42]. Bien que ce ne soit pas si extrême dans le cas de la *Lexikon-Sonate*, le fait qu'un programme informatique soit présenté comme un morceau de musique, renforcé par la référence à la forme historique de la sonate, questionne effectivement les délimitations traditionnelles de l'art. Elle affirme — comme beaucoup d'autres créations contemporaines — la possibilité de nouvelles acceptations de l'œuvre qui enrichissent la tradition artistique plutôt que de nécessairement rompre avec elle.

Une autre spécificité de la *Lexikon-Sonate* découlant de sa forme informatique est l'absence de partition. L'enjeu de la notation a traversé toute l'histoire de la musique et trouve dans cette pièce un écho inédit. La partition classique, à l'origine très simple et conçue comme un simple aide-mémoire, a été sans cesse enrichie à travers les siècles, avec la volonté permanente d'une part de décrire de plus en plus précisément les paramètres d'interprétation, d'autre part de s'affranchir de ses contraintes (la notation du rythme sous formes de noires, croches, etc., est par exemple très contrai-

gnante). C'est pourquoi de nombreux compositeurs du XX^e siècle ont élaboré de nouveaux systèmes de notation, comme Stockhausen qui en créait un spécifiquement pour chaque nouvelle pièce. La figure 17 propose quelques exemples de *partitions graphiques* contemporaines. Renoncer à la partition présente l'avantage de pouvoir explorer la musique sous un nouvel angle (typiquement pouvoir manipuler d'autres hauteurs de son que les seules douze notes de la gamme chromatique), mais revient aussi à renoncer à décrire l'œuvre ainsi créée d'une manière intelligible par n'importe quel musicien. Les conditions d'exécution sont alors beaucoup plus difficiles à réunir. Ce n'est pas le cas avec la *Lexikon-Sonate*, dont la notation est le programme lui-même, qui présente l'avantage d'être un système simple et dépourvu d'ambiguïté pour son « interprète » : l'ordinateur.

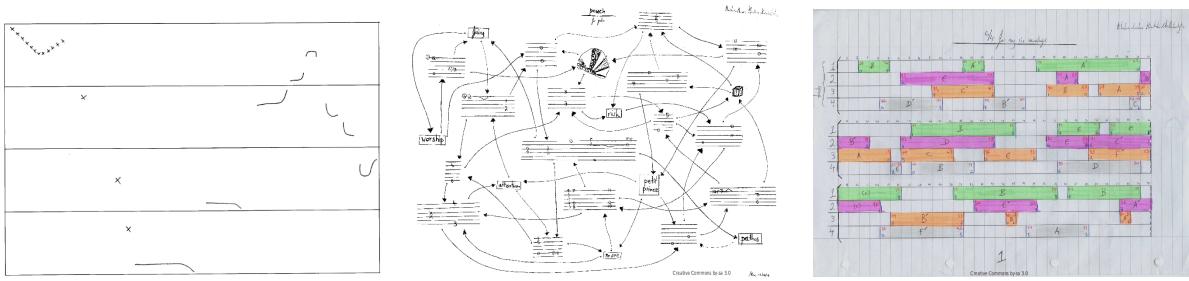


FIGURE 17 – Trois exemples de partitions graphiques, tirées des travaux du compositeur contemporain grec Psimikakis-Chalkokondylis (1989-), respectivement des extraits de *Find the Sound* (2010), *Pouch* (2008) et *6-4* (2009). Ces partitions illustrent les nombreuses recherches des compositeurs à partir du XX^e siècle de nouvelles façons de noter la musique. © Nikolaos-Laonikos Psimikakis-Chalkokondylis.

Le fait que la *Lexikon-Sonate* se passe d'interprète à d'autres conséquences. En particulier, la composition étant affranchie des contraintes de l'instrument, une nouvelle « virtuosité » est possible. Par exemple, un pianiste humain ne peut jouer de *glissando* que sur les seules touches blanches ou les seules touches noires. Dans la *Lexikon-Sonate*, le module GLISSANDI généralise complètement cette définition en générant n'importe quelle suite monotone de notes séparées d'au plus un ton. De même, la figure 18 montre un extrait de la musique générée par le programme telle qu'elle pourrait être reportée sur une partition traditionnelle. Dans les deux cas, le résultat n'est pourtant pas complexe à l'audition ; simplement, la musique peut être déterminée grâce à des modèles qui n'ont plus à tenir compte des limitations physiques de l'instrument et de l'interprète. Cette liberté dans la composition n'est bien sûr pas spécifique à cette

pièce ni nouvelle. Elle est permise par la machine mais ne nécessite pas le temps réel. La nouveauté que l'on trouve ici est à nouveau la synthèse opérée par la composition en temps réel : la composition algorithmique permet de prendre une partie de la responsabilité du compositeur, l'automatisation (de même que tous les procédés d'enregistrement) permet de se passer d'interprète, et dans l'exécution de la *Lexikon-Sonate*, tous deux sont absents. En outre, cela permet de rendre infini ce qui ne pouvait être auparavant que fini. Lorsque la composition nécessitait plus de temps que l'exécution, les pièces devaient être délimitées dans le temps. Ici, composition et exécution sont simultanées : c'est grâce à cela — et bien sûr au fait que l'exécution est confiée à une machine, qui « ne se fatigue jamais »⁷² — que la musique peut se renouveler perpétuellement.

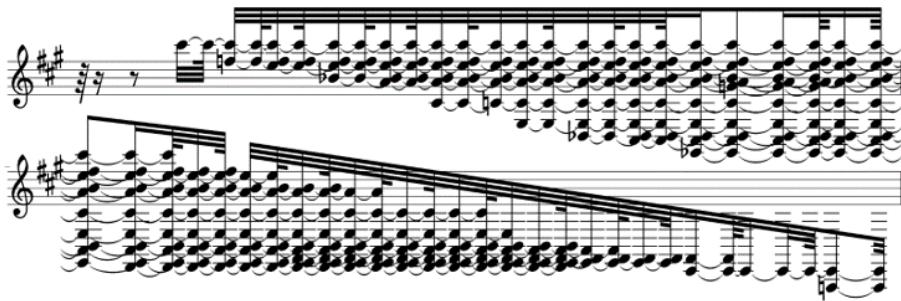


FIGURE 18 – Environ deux secondes de musique jouée par la *Lexikon-Sonate* transcrives sur une partition traditionnelle. Le son produit est simple (un arpège descendant avec les notes tenues), mais la partition paraît extrêmement compliquée. Cela est simplement dû à la liberté rythmique qui ne trouve pas d'équivalent dans ce système de notation. Ici c'est donc surtout une prise de liberté (et non une complexification de la musique) qui est permise par l'ordinateur.

Enfin, cette redéfinition du schéma classique compositeur-interprète-auditeur a pour conséquence une nouvelle forme de sociabilisation⁷³ autour de l'œuvre d'art. D'abord, grâce à l'ouverture de la *Lexikon-Sonate*, la hiérarchie entre l'artiste et son audience est en partie effacée. Essl écrit ainsi : « Je veux mettre l'auditeur au défi de ne pas simplement consommer la pièce, mais tout en l'écoutant de devenir une sorte

72. « Un ordinateur ne se fatigue jamais. Avec l'ordinateur, on peut mettre en scène une musique qui ne s'arrête jamais. Cela n'est évidemment pas possible avec des musiciens. » [Essl et Pagano, 2009] (« *Ein Computer wird nie müde. Mit dem Computer kann man eine Musik in Szene setzen, die nie aufhört, das geht mit Musikern natürlich nicht.* »).

73. D'autres remarques sur les aspects sociaux du projet d'Essl, notamment à travers sa communication sur le réseau Internet, font l'objet de la section 3.5.

de co-créateur, un partenaire du compositeur et de la composition elle-même »⁷⁴ [Essl et King, 1997]. Cette vision idéalise les faits : l'autorité du compositeur reste importante, mais l'auditeur est effectivement invité à approcher la pièce de manière plus active et ludique en tirant profit de l'interactivité (limitée) du programme. Cette interactivité oriente en outre vers une écoute « savante », ou du moins limite la possibilité d'une écoute véritablement naïve, puisque l'utilisation des appels de modules entraîne nécessairement une connaissance de base du fonctionnement de la pièce. Par ailleurs, l'absence d'interprète permet, comme la musique enregistrée, de diffuser plus facilement la *Lexikon-Sonate*, et donc de toucher un public potentiellement plus large, d'autant plus que la pièce est proposée en téléchargement gratuit. Mais les différents auditeurs-utilisateurs auront chacun une expérience inévitablement différente, ce qui fait qu'ils auront moins en commun que s'ils avaient écouté la même musique fixée.

En conclusion, l'analyse de la *Lexikon-Sonate* permet d'illustrer une application de la composition en temps réel, et ses conséquences : problématique d'œuvre ouverte, remplacement de la partition par le programme, libertés compositionnelles supplémentaires, nouvelle relation à l'auditeur. Les modules étudiés dépeignent la manière de programmer d'Essl, en combinant quelques fonctions de la *RTC-lib* qui grâce au recours permanent au hasard deviennent des *Strukturgeneratoren*, avec une recherche fondée davantage sur le résultat musical que sur la complexité des algorithmes.

3.4 Aspects matériels du « hasard » et du « temps réel »

Cette partie aborde dans leur dimension technologique deux notions centrales dans les œuvres d'Essl : le hasard et le temps réel. Nous y expliquons comment l'informatique, et *a fortiori* l'œuvre d'Essl, font une approximation de ces deux phénomènes, en réalité les simulent, d'une manière suffisante pour ses besoins artistiques. Nous y discutons en outre les conditions historiques de l'apparition de la « composition en temps réel ».

74. « *I want to challenge the listener not just to consume the piece but by listening becoming something like a co-creator, being a partner of the composer and the composition itself.* »

3.4.1 Le « hasard »

Le hasard a comme on l'a vu une longue histoire dans le monde musical, bien que l'explosion de son utilisation n'ait eu lieu qu'au XX^e siècle. Pour Essl, il s'agit par conséquent d'un élément déjà étudié et exploité par ses prédecesseurs, qu'il utilise donc comme une possibilité connue du langage musical plus que comme une nouveauté ou une curiosité.

La raison pour laquelle le hasard a pris une telle importance dans l'art de XX^e siècle doit évidemment beaucoup à l'ordinateur⁷⁵, qui permet d'automatiser les processus aléatoires. Il n'est plus nécessaire de lancer des pièces ou des dés, il suffit de demander à l'ordinateur de générer un nombre aléatoire. Cette génération n'est cependant pas réellement aléatoire, car l'ordinateur n'incorpore aucun élément imprévisible comme le dé : il est entièrement déterministe. C'est pourquoi on utilise des générateurs dits *pseudo-aléatoires*, dont l'histoire est aussi longue que celle des ordinateurs du fait des applications en cryptologie ; le premier de ces générateurs est ainsi mis au point par John Von Neumann (1903-1957), l'un des pionniers de l'informatique, l'année même de la mise en service de l'ENIAC, en 1946⁷⁶. Les générateurs pseudo-aléatoires sont des algorithmes qui génèrent des suites de nombres à partir d'une valeur initiale donnée, la *graine*, qui paraissent être le fruit du hasard, ou possèdent des propriétés statistiques communes avec des expériences aléatoires. Le prochain nombre pseudo-aléatoire fourni par un générateur est imprévisible *a priori* pour ses utilisateurs, mais qui a la connaissance de l'algorithme et de la graine peut le calculer, autrement dit le prédire avec certitude. Aux débuts de l'informatique, la génération « rapide » de nombres pseudo-aléatoires était un véritable enjeu, mais que l'augmentation de la puissance des machines à calculer a très rapidement permis de dépasser en ce qui concerne leur utilisation dans le champ artistique.

La simulation du hasard est l'un des principaux attraits de l'informatique pour les compositeurs pionniers de l'informatique musicale, à partir de la fin des années 1950. Par exemple Koenig, qui utilisait déjà des processus aléatoires manuels dans ses compositions, les automatise pour construire les séries qui servent de matériau dans *Projekt*

75. Mais pas seulement, John Cage ayant par exemple utilisé dans ses compositions de nombreuses opérations aléatoires manuelles.

76. Voir à ce sujet [Von Neumann, 1951].

1 (1964). Les générateurs aléatoires programmés par Essl sont les descendants des fonctions écrites par Koenig. On peut aussi prendre l'exemple de l'*ILLIAC Suite*, représentant de la *musique stochastique*. La musique stochastique, initiée et développée par Xenakis, repose sur la structure de chaîne de Markov. Le principe est de pré-établir une liste d'éléments musicaux, ainsi que la probabilité d'enchaîner deux éléments donnés. On fait ensuite appel au hasard — manuel ou automatisé — pour générer un cheminement entre les différents éléments qui obéit aux probabilités données. La figure 19 montre un exemple simple de chaîne de Markov⁷⁷.

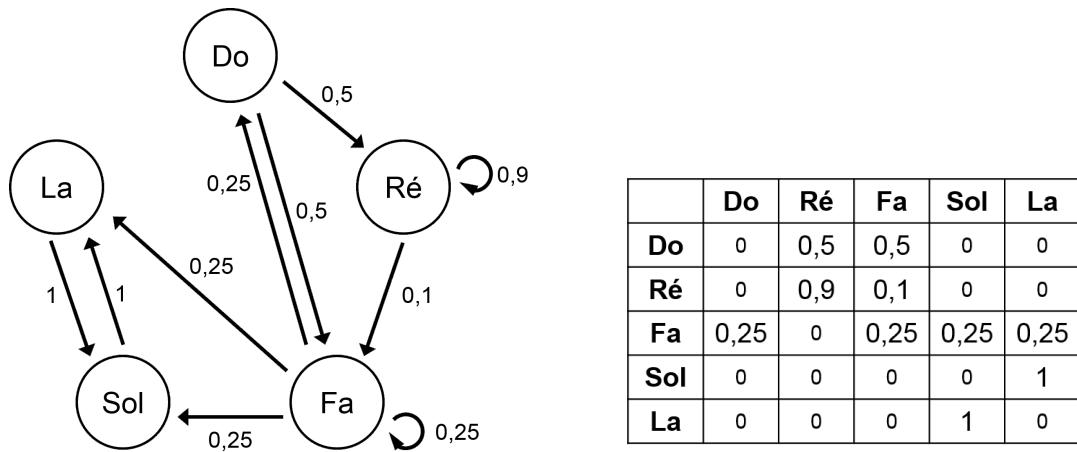


FIGURE 19 – Un exemple simplifié de chaîne de Markov telle qu'utilisée dans la musique stochastique, représentée sous forme de graphe et de tableau (les probabilités sont données entre 0 et 1, sur le graphe les transitions de probabilité nulle ne sont pas dessinées). Le matériau représenté consiste simplement en cinq notes d'une gamme pentatonique (do-ré-fa-sol-la). Il y a une chance sur deux pour que la note suivant un do soit un ré et une chance sur deux pour que ce soit un fa. La note suivant un ré est un autre ré avec probabilité 0,9, et un fa sinon. Après un fa, on peut avoir un autre fa, un do, un sol ou un la de manière équiprobable. La seule note pouvant suivre un sol est un la et réciproquement. La structure étant définie, on peut parcourir le graphe en utilisant des processus aléatoires, afin d'obtenir une suite de notes (par exemple, en partant d'un do : do-ré-ré-ré-fa-do-fa-la-sol-la-sol-la-...). Ce sont des mécanismes similaires qui régissent la musique stochastique comme l'*ILLIAC Suite* ou de nombreuses pièces de Xenakis.

Entre les jeux de dés musicaux et les opérations informatiques pseudo-aléatoires, il y a donc une régression qualitative dans l'utilisation du hasard. L'élément aléatoire est seulement simulé. Cela montre qu'Essl, comme la plupart des artistes, ne s'intéresse pas au hasard de manière ontologique, scientifique, ou métaphysique, mais qu'il ex-

77. Pour une description détaillée de la musique stochastique, voir [Xenakis, 1981].

ploite ses propriétés esthétiques dans ses compositions. Ce qui lui importe, c'est que les phénomènes sonores soient imprévisibles par l'auditeur, un hasard *sensible* et non théorique. L'avantage de l'ordinateur qui prend le pas est sa capacité à simuler le hasard très rapidement. À partir de l'exemple de la figure 19, on pourrait écrire des pièces en jetant des dés ; mais un ordinateur permettrait de le faire *en temps réel*, ou encore de générer un grand nombre de séquences différentes. La présence dans la *RTC-lib* d'un générateur de graines pour les fonctions aléatoires, [RandomSeeder] (qui utilise la date et l'heure), atteste de la maîtrise de ce pseudo-hasard par Essl.

Le chercheur en informatique Philippe Codognet distingue deux utilisations du hasard à des fins esthétiques : le *hasard interne*, ou recours au hasard pendant la composition de l'œuvre, et le *hasard externe*, qui correspond aux éléments imprévisibles lors de son exécution [Assayag et al., 2009, p. 160] ; la meilleure illustration de hasard externe est à cet égard la pièce 4'43" de John Cage. Comme la composition en temps réel fait coïncider composition et interprétation, on pourrait attendre une rencontre de ces deux hasards. Mais le hasard externe n'apparaît qu'au travers de certaines possibilités interactives, généralement limitées, des pièces composées par Essl, ou dans les performances improvisées qu'il donne. C'est bien le hasard interne qui est prépondérant dans les fonctions de la *RTC-lib* et des pièces comme la *Lexikon-Sonate*.

3.4.2 Le « temps réel »

La « composition en temps réel » étant la notion la plus remarquable de l'œuvre d'Essl reposant sur le progrès technologique, il est intéressant d'étudier précisément les conditions historiques qui ont permis son apparition.

Il convient d'abord de préciser que, comme pour le hasard, il n'y a pas de « temps réel » au sens d'une immédiateté dans l'interaction entre l'homme et la machine. Comme le rappelle le compositeur Philippe Manoury (1952-) : « C'est en fait une illusion. Le temps réel n'existe jamais dans la réalité technologique parce qu'une machine met toujours un certain temps, même si celui-ci est extrêmement bref, pour effectuer ses calculs. [...] En musique, on parle de temps réel à partir du moment où le laps de temps entre le début du calcul et la livraison du résultat d'une opération informatique est suffisamment bref pour ne pas être perçu. » [Manoury et al., 2012, p. 41-42]. Ainsi, l'accent

est à nouveau mis sur les perceptions de l'auditeur : ce qui est suffisamment bref pour sembler sans durée est appelé temps réel.

C'est en 1991 qu'Essl découvre la possibilité du temps réel, grâce à la Station d'informatique musicale de l'IRCAM. Cette « station » est simplement un ordinateur central ou *mainframe*, construit à partir d'ordinateurs NeXT et d'une carte son spécialisée. L'interaction en temps réel est une nouveauté considérable pour Essl par rapport à ce qu'il connaît sur son Atari ST personnel : « En LOGO, il fallait de longues heures pour calculer une liste de données que je devais ensuite transformer en notation musicale pour pouvoir l'analyser— une procédure qui prenait un temps considérable. » [Essl, 2010]. La figure 20⁷⁸ montre que la puissance de la Sim était effectivement bien supérieure à celle de l'Atari ST⁷⁹. À cette époque, la génération de musique ou le traitement du son en temps réel étaient donc possibles sur des ordinateurs de type *mainframe* mais pas sur les micro-ordinateurs : il s'agit du moment précis où le progrès technologique, incrémental, a permis leur réalisation, véritable progrès qualitatif.

La différence entre les calculs de plusieurs heures d'Essl et le temps réel à l'IRCAM s'explique par deux facteurs supplémentaires. D'abord, Essl programmait avec le langage LOGO, et même avec une version interprétée de ce langage, nommée xLOGO, ce qui signifie que l'exécution des codes qu'il écrivait nécessitait une étape de « traduction » vers un langage informatique plus simple. Ce langage offrait aussi une bien meilleure expressivité et des fonctionnalités plus étendues que, par exemple, le BASIC ; en contre-partie, des codes simples peuvent correspondre à un nombre important d'instructions machine élémentaires. La Sim, en revanche, utilisait le langage MAX, développé expressément pour elle et optimisé pour les tâches de musique algorithmique. En outre, même avec un langage plus performant et un ordinateur plus puissant, Essl aurait dû traduire les résultats de ses programmes en notation musicale. Le progrès matériel que représente la Sim réside dans l'association d'un ordinateur puissant avec une *carte son*, c'est-à-dire un composant électronique dédié à l'interface entre l'unité de calcul et des périphériques audio comme des hauts-parleurs. Celle-ci peut prendre

78. D'après <http://doornbusch.net/> et https://en.wikipedia.org/wiki/Instructions_per_second#MIPS (consultés le 30 août 2016).

79. Plus précisément, il s'agit des puissances des processeurs utilisés par ces ordinateurs, respectivement trois cartes avec deux Intel i860 de 1991 pour la Sim et le Motorola 68000 de 1982 pour l'Atari ST

Année	MIPS	Modèle	Type
1954	0,000640	IBM 704	<i>Mainframe</i>
1969	3,3	IBM 360/85	<i>Mainframe</i>
1973	0,065500	DEC PDP 11/45	Micro
1977	0,230	Apple II	Micro
1985	2,188	Atari ST	Micro
1988	6×50	Sim (IRCAM)	Mainframe
1997	525	PowerBook G3	Micro
2003	3100	iMac G5	Micro
2008	50000	Apple Mac Pro	Micro

FIGURE 20 – Tableau indiquant la puissance de quelques modèles d’ordinateurs, en millions d’instructions par seconde (MIPS) exécutables par le processeur. Il est précisé pour chacun s’il s’agit d’un *mainframe* (ou *ordinateur central*, machine centralisée de grande taille) ou d’un micro-ordinateur. Les modèles en gras correspondent à ceux utilisés par Essl dont il est question dans cette partie, les autres sont donnés à titre indicatif.

en charge une grande partie des calculs spécifiques au traitement du signal, grâce à quoi la conversion en son pouvait avoir lieu en « temps réel ». De l’analyse de la *RTC-lib* (section 3.2), concluant que la plupart des fonctions étaient d’une complexité algorithmique faible, on déduit que l’aspect compositionnel de la « composition en temps réel » n’exploitait même pas toutes les capacités de la Sim (et nécessite une puissance de calcul très en-deçà de celle des ordinateurs standards actuels). L’avancée déterminante pour permettre l’existence de cette notion se situait dans le traitement sonore *via* la carte son, et non dans la puissance de calcul brute.

La figure 20 montre que, du fait de l’augmentation rapide de la puissance des ordinateurs, l’Apple PowerBook G3 acheté par Essl en 1997 atteint et même dépasse les capacités de la Sim. Cette croissance est exponentielle : elle suit approximativement la loi empirique de Moore⁸⁰, qui affirme que le nombre de transistors sur un micro-processeur pouvant être fabriqué à un coût donné double tous les ans environ. C’est pourquoi de nombreux progrès ont lieu sur de très courtes échelles de temps. En ce qui concerne le « temps réel » musical, il apparaît donc que les prototypes comme la

80. Décrise dans [Moore, 2000].

Sim l'ont concrétisé dès qu'il a été technologiquement possible, de même qu'Essl dans un second temps, lorsqu'il a été accessible aux micro-ordinateurs.

3.5 Un compositeur de l'ère numérique

Cette dernière partie traite d'un aspect peu abordé jusqu'ici du rôle de l'ordinateur dans le travail d'Essl, l'utilisation qu'il fait du réseau Internet. Les enjeux en sont moins technologiques que sociaux. La façon dont ce réseau mondial, qui est l'un des piliers de l'informatique au sens large et le principal moteur de la culture numérique, bouleverse l'activité du compositeur, mérite d'être étudiée. Internet ouvre de nouvelles perspectives à la fois dans le rapport de l'artiste à l'œuvre, au public, et à ses homologues⁸¹. Chez Essl, la redéfinition du rapport à l'œuvre passe avant tout par la question de l'auctorité. Celle du rapport au public réside dans l'utilisation d'Internet comme medium de masse, moyen de diffusion efficace au service d'une volonté de démocratisation. Enfin, la communication joue un rôle important pour Essl en particulier parce qu'elle lui permet d'entrer en contact et de collaborer avec d'autres artistes.

De même que l'apparition des micro-ordinateurs et leur démocratisation, Essl a vécu leur mise en réseau qui a conduit à l'avènement d'Internet. L'origine de cette « Toile » permettant de relier n'importe quels ordinateurs à l'échelle mondiale remonte aux années 1960. Arpanet, lancé en 1969, premier réseau à grande échelle reposant sur le principe de *transfert par paquets*, est considéré comme son premier ancêtre. Les techniques d'échange de données entre ordinateurs évoluent rapidement et les réseaux se succèdent, en particulier Bitnet à partir de 1981, qu'Essl a connu pendant ses études [Ehrler, 1999], et Internet depuis 1989. Ces réseaux permettent aux ordinateurs de servir d'interface pour l'échange de données à grande distance, symbolisé par le courrier électronique, inventé dès 1965.

La vitesse des communications et l'effacement des barrières géographiques qu'elle entraîne modifient radicalement les perspectives traditionnelles de circulation des informations. Ils font par exemple apparaître le problème de la surveillance et de la pro-

81. Ces rapports étudiés ici sont ceux qu'Essl entretient pendant la partie la plus récente dégagée dans son parcours, celle où il devient compositeur-performeur plutôt que de travailler « enfermé dans une tour d'ivoire ».

tection de la vie privée, ou celui du téléchargement illégal. Dans le cas d'Essl, comme de tout artiste, c'est justement la question du statut auctorial et de la distribution de ses œuvres — partitions, enregistrements, programmes — qui est posée par le monde numérique. Sa position est intéressante. D'un côté, il cède une grande partie des priviléges traditionnels de l'auteur. Ainsi distribue-t-il la plupart de ses programmes⁸² sous licence libre, comme nous l'avons vu dans le cas de la *Lexikon-Sonate* et dans celui de la *RTC-lib*, qui sont pourtant respectivement son œuvre au plus grand succès et le socle logiciel de tout son travail algorithmique de composition. De même, Essl rend ses partitions disponibles et gratuites. Il met à disposition sur son site personnel de nombreux enregistrements de ses pièces, ainsi que sur sa chaîne YouTube où il rend publiques des performances, des entretiens et des cours (plus de 300 séquences vidéo à ce jour). Ce parti-pris s'explique grandement par le contexte historique. Comme cela a été mis en avant dans la partie 2.2, Essl fait partie de la génération qui a connu la généralisation des micro-ordinateurs et l'idéal de liberté à laquelle elle s'associe. À la même époque, dans le courant des années 1980, apparaît en outre le courant du logiciel libre⁸³, qui promeut le partage des ressources logicielles de tous vers tous et gratuitement, auquel Essl s'associe donc dans la distribution de ses travaux que nous venons d'évoquer. Chacun de ses programmes téléchargeables inclut ainsi une licence délimitant précisément les usages que peut en faire l'utilisateur, ce qui montre la préoccupation du compositeur sur cette question. Toutefois, il affirme dans le même temps son statut auctorial ainsi que sa propriété intellectuelle, son droit d'auteur. Il ne s'oppose pas à ce que certaines exécutions de ses œuvres soient payantes, touchant des droits sur les concerts ou CD dans lesquels d'autres musiciens jouent ses compositions. Il lui arrive aussi, comme nous l'avons vu dans le cas de la *Lexikon-Sonate*, de conserver une version améliorée de ses programmes exclusivement pour ses propres performances ; dans le cas de son instrument *m@ze* 2, il n'existe même pas de version publique. Un événement dans le développement de la *Lexikon-Sonate* est révélateur de l'intérêt porté par Essl à la question auctoriale : à l'origine, le programme possédait

82. Seuls le *R Eplay PLAYer* et *fLOW* sont des *sharewares*, c'est-à-dire qu'il en existe deux versions, l'une gratuite mais aux fonctionnalités limitées, l'autre complète mais payante.

83. Plus précisément, ce sont les logiciels propriétaires qui apparaissent, le logiciel étant auparavant considéré comme un objet scientifique et circulait comme tel dans le milieu universitaire. Pour plus de détails sur l'histoire du logiciel libre, voir [Paloque-Berges et Masutti, 2013].

une sortie MIDI, mais Essl a découvert que « des personnes l'ont utilisé pour générer des partitions. Comme si c'était un programme générateur de partitions. Et ils ont ainsi créé des morceaux qu'ils ont vendu comme leurs propres morceaux »⁸⁴. Essl a donc désactivé la sortie MIDI de la version publique, consolidant ainsi sa position d'auteur.

L'une des spécificités du réseau Internet est qu'il « met en jeu un type de liaison multimodale entre les utilisateurs sans précédent : la liaison de *tous vers tous* [où] chaque internaute est à la fois récepteur et émetteur » [Couchot et Hillaire, 2003, p. 63]. Remarquons avant tout qu'Essl n'exploite cependant pas tout ce potentiel : si, comme nous allons le voir, il tire profit de la liaison de *un vers tous* pour la diffusion de son œuvre, il a peu recours à celle de *tous vers un*, c'est-à-dire les possibilités d'interactivité. En ce sens, son utilisation du réseau mondial est la même que celle des media de masse (radio, télévision, cinéma), dont les propriétés sont simplement amplifiées par Internet. Cette limitation de l'interactivité est d'ailleurs une caractéristique générale de l'œuvre d'Essl. Contrairement à beaucoup d'autres compositeurs, il n'en fait pas un enjeu esthétique important ; ses programmes et installations permettent une interaction mais l'utilisateur a des possibilités restreintes plutôt qu'une liberté totale. Même son instrument *m@ze 2* n'est pas interactif au sens large : bien que certaines de ses fonctionnalités aient recours au hasard, celles-ci reposent toujours sur l'idée de *Strukturgenerator*, et produisent donc une réponse imprévisible mais jamais imprévue aux stimuli qu'il reçoit.

On peut identifier dans la démarche d'Essl une véritable tentative de démocratisation de son travail, de le rendre accessible à la fois en termes intellectuels et logistiques. Dans sa démarche esthétique, ce pas vers le public passe par le refus d'une musique trop abstraite⁸⁵, l'utilisation de matériaux musicaux connus de tous⁸⁶, ou d'une

84. Entretien en annexe.

85. « Je ne fais pas de la musique abstraite qu'on ne peut comprendre que si l'on a une formation spéciale, ou des connaissances particulières, ou beaucoup d'expérience. Ce qui m'intéresse, c'est avant tout ce qu'il y a d'immédiat, d'extatique, qui se révèle à celui qui est prêt à s'abandonner. » [Essl et Kühnelt, 2006/2007] (« *Ich mache keine abstrakte Musik, die man nur versteht, wenn man eine spezielle Ausbildung hat, oder Wissen oder viel Erfahrung. Mir geht es vor allem um das Unmittelbare, das Ekstatische, das sich demjenigen erschließt, der bereit ist sich dem hinzugeben.* ») ; voir aussi [Essl et Günther, 1998].

86. « Je voulais délibérément utiliser un matériau très succinct, qui soit de plus facile à identifier. [...] Il existe ainsi certaines musiques très caractéristiques, comme aussi la *Cinquième* de Beethoven ; [...] il suffit de les jouer et chacun sait immédiatement de quoi il s'agit. » [Essl et Huber, 2014] (« *Ich wollte*

manière générale l'attachement au son et à l'effet qu'il produit à l'intérieur de la démarche algorithmique⁸⁷. Sur le plan pratique, il s'incarne dans la recherche de lieux de concerts non-traditionnels comme les musées, en particulier le Essl Museum, et naturellement dans l'utilisation d'Internet pour diffuser son œuvre. Les aspects de « liberté » déjà évoqués dans la diffusion d'enregistrements et de programmes y participe. De plus, Essl multiplie les lieux virtuels de sa présence sur Internet : les réseaux sociaux Twitter et Facebook, les plate-formes de distribution audio et vidéo SoundCloud et YouTube, ainsi que sa page personnelle⁸⁸. Celle-ci fait office à la fois de vitrine [Essl et Pagano, 2009] et de musée — « L'Internet apparaît [...] comme le lieu de [la] mémoire à venir, comme musée virtuel global. » [Couchot et Hillaire, 2003, p. 231]. Essl y propose, outre des liens pour télécharger ses programmes ou consulter ses séquences vidéos et enregistrements, sa propre actualité (compositions, concerts), de nombreux éléments biographiques, une liste exhaustive de ses écrits, et un recensement des publications à son propos. Cela représente un travail considérable qui lui permet de faire sa propre muséification. On peut cependant se demander si cette démarche n'est pas une condition nécessaire de visibilité donc d'existence dans un monde globalisé, saturé de communications et soumis à une très forte concurrence, y compris dans le domaine artistique. De même, on peut s'interroger sur la portée de la démocratisation qu'elle vise, car comme le soulignent Hillaire et Couchot : « L'ambitieuse volonté d'une communication universelle est contrecarrée par la réalité. Le public attendu, innombrable, multi et transculturel, reste encore un public de spécialistes très liés au monde de l'art et très identifiables socialement. Les "communautés virtuelles" sont en fait des microcosmes assez fermés, avec déjà leurs traditions et leurs orthodoxies. »

ganz bewusst ein sehr knappes Material verwenden, das auch leicht wiedererkennbar ist. [...] Es gibt bestimmte signalhafte Musiken, wie auch die Fünfte Beethovens [...] Das heißt, das braucht man nur anspielen und jeder weiß sofort, was das ist. »).

87. « Je crois que l'algorithme ne doit jamais devenir un fétiche se suffisant à lui-même, à la logique duquel il faudrait faire appel pour dire : cette structure est si belle, si mathématique à tous points de vue, si cohérente. Cela doit toujours passer à l'épreuve de la réalité et de notre expérience d'auditeurs. » [Förster, 2011] (*« Ich glaube der Algorithmus darf nie ein sich selbst genügender Fetisch werden, auf dessen Logizität man sich beruft und sagt : Diese Struktur ist so schön und alles ist mathematisch in Beziehung und damit stimmig. Es muss immer an der Wirklichkeit und an unserer Hörerfahrung getestet sein. »*).

88. <http://www.essl.at> (consultée le 30 août 2016) sur laquelle on trouvera quelques remarques dans [Ehrler, 1999].

[Couchot et Hillaire, 2003, p.79].

Enfin, la dimension communicationnelle d'Internet joue, d'après Essl⁸⁹, un rôle crucial dans son parcours. Il s'agit en l'occurrence bien d'une liaison *tous vers tous*, qui implique non pas le public mais d'autres artistes. Comme cela a été expliqué dans la partie 2.4, Essl a donné une orientation nouvelle à sa carrière de compositeur après sa présentation au Festival de Salzbourg de 1997, en ouvrant sa pratique, et en particulier en collaborant avec d'autres musiciens⁹¹. Dans cette optique, sa page personnelle peut ainsi être comprise non seulement comme un moyen de visibilité envers le public mais aussi, avec la facilité de contact qu'elle propose, une invitation tacite à la collaboration pour les artistes qui la consulteraient. L'activité soutenue d'enseignement menée par Essl, à l'Académie de musique et des arts du spectacle de Vienne, mais aussi en tant que professeur invité dans des universités du monde entier, et dans les programmes « pédagogiques » [Klein, 2013] qu'il propose au Essl Museum, souligne aussi l'importance que revêt pour lui la communication. On peut donc dire que l'avènement du réseau Internet a fait d'Essl, comme de tant d'autres, un citoyen global, et qu'il a immédiatement tiré profit des possibilités ouvertes par ce nouveau medium et en a fait un enjeu central de sa pratique artistique, la rendant plus collaborative.

Ces quelques remarques ébauchent la façon dont Essl utilise au compte de sa pratique artistique la révolution Internet. Le réseau est avant tout utilisé comme un medium, c'est-à-dire un outil de diffusion d'information et de communication. Il est mis au service de l'œuvre, en particulier de sa visibilité, mais sans sacrifier l'image de l'auteur. Essl ne s'efface pas derrière ses créations, et affirme au contraire son individualité.

89. « D'après mon expérience personnelle, je pense que [l'aspect spécifique d'Internet] le plus important est la communication. »⁹⁰ [Essl et Föllmer, 2004].

91. « À travers le contact avec des pairs partageant mes goûts, qui avait d'abord lieu principalement grâce à des *mailing lists*, j'ai mûri de nouvelles idées. [...] Le caractère non-hiéarchique d'Internet, avec sa structure rhizomatique, possède de multiples points communs avec ma propre vision d'une musique comme un réseau de communication dont on peut faire l'expérience sensible » [Essl et Föllmer, 2004] (« *Durch den Kontakt mit Gleichgesinnten, der zunächst hauptsächlich über Mailing-Lists ausgetragen wurde, reiften in mir neue Ideen [...] Der nicht-hierarchische Charakter des Internets mit seiner rhizomatischen Struktur zeigt vielfache Entsprechungen zu meiner persönlichen Sichtweise, Musik als kommunikatives Netzwerk aufzufassen und dieses auch sinnlich erfahrbar zu machen.* »).

3.6 Conclusion : Essl et l'ordinateur

Dans les différentes analyses qui précèdent, le rapport qui se dessine entre Essl et l'ordinateur, et plus généralement le monde numérique, est celui d'un artisan avec son outil. C'est ainsi qu'on a parlé de « boîte à *outils* » à propos de la *RTC-lib* et de la façon dont elle est utilisée dans la *Lexikon-Sonate* ; et on a vu qu'il utilise le réseau Internet avant tout comme un *outil* de diffusion. Comme l'écrit le musicologue Bernhard Günther : « Essl utilise les possibilités techniques actuelles comme un moyen de créer des structures musicales »⁹² [Günther, 1995] : un moyen, et non une fin. L'étude de ses programmes montre bien que c'est l'aspect sensible⁹³ qui a le dernier mot dans le processus de création, qu'Essl cherche à tirer parti de cet outil informatique pour donner vie à des idées musicales, et non à programmer des structures complexes pour leur donner ensuite un sens musical. Ce rapport à l'ordinateur comme outil transparaît aussi dans le modèle qu'utilise Essl : un PowerBook, à la fois moins performant que les appareils standards actuels, et antinomique de sa philosophie du libre. Il le reconnaît lui-même⁹⁴, justifiant ce choix par le fait qu'il est « plus productif » avec cet ordinateur auquel il est habitué. En outre, les analyses de la *RTC-lib* et de la *Lexikon-Sonate* ont montré à la fois qu'Essl possédait une véritable maîtrise de cet outil bien qu'il ne soit pas spécifique au domaine musical, et qu'il subissait aussi ses contraintes intrinsèques.

Il en résulte un rapport de force entre la technologie et le compositeur. Essl affirme sa singularité d'artiste et refuse que la technologie prenne le pas sur l'art. La limitation de l'interactivité dans ses créations ou encore le rejet des techniques d'intelligence artificielle pour la composition⁹⁵ montrent qu'il est musicien avant d'être programmeur. Bien que les algorithmes bénéficient d'un statut de premier plan dans son œuvre, ils

92. « *Essl nutzt die heutigen Möglichkeiten der Technik als Mittel, musikalische Möglichkeitsstrukturen zu schaffen.* »

93. Se ramener exclusivement aux sens, comme l'affirme Essl, est exagéré : il serait plus juste de parler de la tradition musicale historique. Si l'écoute apparaît primordiale, les procédés issus du serialisme qui en sont déconnectés ont aussi une place conséquente dans l'œuvre d'Essl. On a cependant conservé ce terme de « sensible » pour marquer l'opposition avec l'expérimentation abstraite des possibilités spécifiques de l'ordinateur.

94. Voir entretien en annexe.

95. Voir entretien en annexe.

ne le sont jamais au détriment des phénomènes sonores.

Il y a cependant une dimension exploratoire dans ce rapport à l'outil informatique, et donc créatif. Essl n'est pas un simple utilisateur, comme ce serait le cas s'il n'exploitait que des logiciels centrés sur un séquenceur, qui définissent nécessairement une orientation esthétique forte⁹⁶, mais s'est approprié des langages de programmation afin de pouvoir développer plus librement ses propres idées. Il affirme même la programmation comme un impératif du compositeur informatique : « Nous devons programmer l'ordinateur nous-même, autrement dit nous devons formuler nos idées compositionnelles propres, personnelles, individuelles, et nos propres interrogations sous la forme de programmes informatiques »⁹⁷ [Förster, 2011]. On a ainsi exhibé à la fois l'exploitation de l'ordinateur pour mettre en œuvre des idées musicales préexistantes, mais aussi l'élaboration de nouvelles formes et notions, en particulier la composition en temps réel, qu'Essl a développée dès le moment où elle a été technologiquement possible.

Le travail d'Essl a ainsi réalisé un progrès manifeste. C'est un progrès esthétique⁹⁸, et non scientifique ou technique — Essl n'a pas par exemple conçu les machines qu'il utilise —, mais fondé sur l'exploitation des possibilités des progrès techniques — dans le cas de la composition en temps réel, l'augmentation de puissance des ordinateurs et dans une moindre mesure le développement du langage MAX. En résumé, il s'agit d'une approche *opportuniste* de la technologie. Essl ne provoque pas les innovations technologiques mais apprend à maîtriser les possibilités disponibles, pour en tirer profit dans son travail artistique.

4 Conclusion

Fasciné comme toute une génération par la démocratisation des micro-ordinateurs et leur mise en réseau à l'échelle mondiale, Karlheinz Essl a pleinement incorporé cette

96. Comme *ProTools*, *Cubase*, *Ableton Live*...

97. « *Wir müssen den Computer selber programmieren, d. h. wir müssen unsere eigenen persönlichen individuellen kompositorischen Ideen und Fragestellungen in Form von Computerprogrammen formulieren.* »

98. La notion de *progrès*, en science et dans le domaine des arts, est employée telle qu'elle est discutée dans [Kuhn, 1972].

nouvelle technologie dans son art. Musique électronique et synthèse sonore, composition algorithmique puis composition en temps réel, installations, méta-instruments, œuvres collaboratives : il n'a cessé d'explorer les nouvelles formes d'expression musicale permises par l'informatique. Son intérêt permanent pour la musique numérique d'avant-garde lui a permis de développer en pionnier, dès le moment de sa possibilité technique, la notion de composition en temps réel, synthèse des méthodes de composition algorithmique et du traitement du signal sonore en « temps réel », qui représente un véritable progrès dans l'exploitation de l'informatique par la musique savante.

L'analyse de ses programmes montre cependant qu'Essl s'inscrit dans la continuité de sa discipline, réfléchissant en termes musicaux et de créativité sans être attiré par les possibilités techniques qui lui semblent sans lien avec la tradition artistique ou les perceptions de l'auditeur. L'ordinateur est pour lui un exécutant rapide d'algorithmes de composition, un générateur de hasard, un interprète infatigable, un vecteur de diffusion et un moyen de communication, en somme un moyen pour parvenir à ses fins artistiques, un outil, central dans sa pratique quoique inféodé à l'artiste.

Dans l'histoire de la musique, Essl apparaît comme l'héritier esthétique de Gottfried Michael Koenig et de l'Institut de Sonologie, place forte du développement de la musique algorithmique. Leur tradition est celle d'une musique générative, où le compositeur élabore l'architecture d'ensemble de ses créations, et où les finitions sont confiées au hasard et à l'ordinateur. L'apport d'Essl a été d'explorer les développements de cette démarche à la lumière des progrès matériels et logiciels de l'informatique. Il s'agit naturellement d'une façon parmi d'autres de donner un sens à la technologie dans le champ musical savant. Elle s'oppose notamment aux recherches esthétiques de Xenakis et de son école, à l'origine par exemple de l'environnement de programmation *IannisX*, chez qui la musique algorithmique a recours à des procédés plus abstraits et plus complexes, qui contrôlent la structure générale des morceaux, ont recours à des outils mathématiques extra-musicaux, et travaillent le son avec des méthodes aléatoires continues plutôt que discrètes.

Le rapport qu'entretient Essl à l'ordinateur est emblématique de sa génération, qu'il s'agisse du rôle de l'informatique dans la musique ou dans la société en général. Ses prédecesseurs, comme Koenig ou Stockhausen, ont été les témoins de la naissance de l'informatique moderne. Maîtrisant déjà la musique électronique, ils ont intégré en

pionniers cette nouvelle science, objet de curiosité et de fantasmes, à la musique savante. Ses successeurs⁹⁹, qui ont grandi dans un monde numérique, déjà informatisé, font de l'ordinateur et de tout ce qui y a trait un objet commun des représentations. C'est ainsi que la pratique du *live coding* propose au public de voir le code informatique au cours de son élaboration, supposant une familiarité du public suffisante pour apprécier les compétences techniques aussi bien qu'artistiques du performeur. Chez Essl, l'ordinateur reste privé et la programmation, hors du temps ; mais l'informatique n'est plus une composante exceptionnelle de l'œuvre, c'est un outil maîtrisé et omniprésent, un élément essentiel, indispensable sans être banal, de la table de travail du compositeur.

99. Nommons à titre d'exemple Thor Magnusson (1972-), développeur de l'environnement de *live coding* *ixi lang*, ou Nick Collins (1975-) et Alex McLean (1975-), inventeurs des *algoraves*, événements dansants autour de musique *live coded*.

Résumé

L'informatique est omniprésente dans l'œuvre du compositeur contemporain autrichien Karlheinz Essl (1960-), qui produit aussi bien des partitions que des programmes, des installations interactives ou des performances avec ordinateur. L'objet de ce document est d'étudier son cas, dans une approche historique, comme un exemple de l'assimilation de l'informatique par l'art contemporain.

Une première partie fait l'étude historique du parcours d'Essl. Celui-ci est décrit en quatre étapes. D'abord, les années d'études du compositeur (jusqu'en 1984), au cours desquelles se forment ses principales influences qui le prédisposeront à s'intéresser à l'informatique. Ensuite (1984-1991), Essl fait ses premiers pas dans ce domaine suite à l'acquisition d'un premier ordinateur personnel, s'initiant à la programmation et intégrant des aspects algorithmiques à ses compositions. Plus tard (1991-1997), il développe ses travaux algorithmiques en leur associant une exécution en « temps réel » : il crée des morceaux générés par ordinateur et joués au même moment, ce qu'il nomme la *composition en temps réel*. Cette démarche qui était auparavant technologiquement impossible est permise par un ordinateur prototypique dédié à la musique construit à l'IRCAM, qu'Essl découvre lors d'un stage en 1991. Un dernier tournant a lieu en 1997, lorsque Essl décide de devenir *performeur* en plus de compositeur, concevant notamment son propre instrument virtuel qu'il utilise dans des performances improvisées.

La seconde partie étudie l'utilisation de l'informatique par Essl et le rôle qu'elle joue dans son œuvre à partir de l'étude de ses programmes. Ces programmes, en particulier la bibliothèque *RTC-lib* qui fournit des fonctions à la plupart de ses morceaux informatisés, et sa pièce emblématique, la *Lexikon-Sonate*, sont considérés en tant que sources historiques. Leur analyse dégage les différents aspects du rapport d'Essl à l'ordinateur, qu'il utilise comme un outil au service d'idées musicales, privilégiant la programmation de fonctions simples et sensiblement intelligibles. La discussion des notions de « hasard » et de « temps réels » en informatique révèle une approche opportuniste de la technologie : Essl exploite les possibilités permises par le progrès technique sans les provoquer, et s'intéresse au résultat sensible plutôt qu'aux aspects techniques sans lien avec sa pratique. Enfin il est montré que ces caractéristiques trouvent un écho dans l'utilisation, d'ordre avant tout communicationnel, que le compositeur fait du réseau Internet.

Mots-clefs :

- Karlheinz Essl
- informatique musicale
- musique algorithmique
- composition en temps réel
- code source
- histoire de l'informatique

5 Bibliographie

Sources primaires

Interviews

- Essl, Karlheinz. « Generative Music ». Questionnaire de Håkon Normann, 2002.
- Essl, Karlheinz, et Hanns Abele. « Komponieren im Cyberspace ». 1998.
- Essl, Karlheinz, et Alice Ertlbauer. « Sonnez la cloche ! » 2003.
- Essl, Karlheinz, et Golo Föllmer. « NET MUSIC ». *Neue Zeitschrift für Musik* 5 (2004).
- Essl, Karlheinz, et Stefan Gfeller. « Algorithmische Komposition und Live-Elektronik ». Mémoire de maîtrise, Hochschule der Künste Bern, 2010.
- Essl, Karlheinz, et Magdalena Halay. « Composing with Algorithms ». 2015.
- Essl, Karlheinz, et Jack Hauser. « Musik-Wissenschaft an ihren Grenzen », 507–516. Peter Lang, 2004.
- Essl, Karlheinz, et Katharina Hötzenecker. « OMNIA IN OMNIBUS : Behind the Scenes ». 2016.
- Essl, Karlheinz, et Annegret Huber. « Wagner in Translation ». In *Performing Translation. Schnittstellen zwischen Kunst, Pädagogik und Wissenschaft*, 238–251. Löcker Verlag, 2014.
- Essl, Karlheinz, et Reinhard Kager. « Amazing maze ». 1997.
- Essl, Karlheinz, et Reinhard Kapp. « Gibt es sie noch : "die" Musik ? » In *ANKLAENGE 2010*. Mille Tre Verlag, 2011.
- Essl, Karlheinz, et Joanna King. « Profile Karlheinz Essl ». 1997.
- Essl, Karlheinz, et Lothar Knessler. « Radioportrait Karlheinz Essl ». Émission Zeit-Ton, 1994.
- Essl, Karlheinz, et Elisabetta Pagano. « Elektronische Musik und Emotion ». In *Experimentieren - improvisieren - zuhören. Ursprung, Entwicklung und Theorien der heutigen experimentellen elektronischen Musik sowie Beweggründe und Gedanken in Wien tätiger Musikschaender*. Südwestdeutscher Verlag für Hochschulschriften, 2009.
- Essl, Karlheinz, et Carl-Marcus Piswanger. « Erfahrungen mit Elektronischer Musik ». *Monitor* (décembre 2008).
- Essl, Karlheinz, Esther Planton et Andrea Lexer. « A-mze-ing ». Campus Musik, 2009.
- Essl, Karlheinz, et Ruth Ranacher. « Mein Instrument ist der Computer ». 2015.
- Essl, Karlheinz, et Wolfgang Rauscher. « OUT OF THE BLUE ». 2011.
- Essl, Karlheinz, et Tim Schaffrick. « Live ? Über das Verhältnis von Komposition und Live-Performance ». 2004.

- Essl, Karlheinz, et Eileen Schreiber. « Algorithmische Komposition im Schaffen Karlheinz Essls ». 2012.
- Essl, Karlheinz, et Christoph Wagner. « Faszination Toy Piano ». 2014.

Articles théoriques

- Essl, Karlheinz. « Algorithmic composition ». *Cambridge Companion to Electronic Music. Cambridge University Press, Cambridge* (2007).
- _____. « Composing in a Changing Society ». In *International musicological symposion Austria*. 1996.
- _____. « Computer Aided Composition ». *Distel* 46/47 (1991).
- _____. « Improvisation with Computers ». Questionnaire de Thomas Peter, 2006.
- _____. « Klangkomposition und Systemtheorie ». In *Darmstädter Beiträge zur Neuen Musik*, t. XX. Südwestdeutscher Verlag für Hochschulschriften, 1994.
- _____. « Komponieren im Cyberspace ». In *Kultur als Kompetenz. Neue Technologien, Kultur & Beschäftigung*. 1998.
- _____. « Kompositorische Konsequenzen des Radikalen Konstruktivismus ». *Positionen. Beiträge zur neuen Musik* 11 (1992) : 2–4.
- _____. « New Aspects of Musical Material ». Leçon donnée aux Darmstädter Ferienkurse für Neue Musik, 1992.
- _____. « Plädoyer für "Das Offene Kunstwerk" ». *Positionen. Beiträge zur neuen Musik* 26 (1996) : 56–59.
- _____. « sine fine... Unendliche Musik ». *POSITIONEN. Beiträge zur Neuen Musik* (2008) : 41–43.
- _____. « Strukturgeneratoren — Algorithmische Komposition in Echtzeit ». *Beiträge zur Elektronischen Musik* 5 (1996).
- _____. « Wie entsteht ein Komposition ? » In *Bericht des Bruckner-Symposion 1995 - "Zum Schaffensprozeß in den Künsten"*. Anton Bruckner Institut Linz, 1997.
- Essl, Karlheinz, et Bernhard Günther. « Realtime Composition : Musik diesseits der Schrift ». *Positionen. Beiträge zur neuen Musik* 36 (1998) : 4–9.

Œuvres

- Essl, Karlheinz. « Lexikalische Metamorphosen ». In *Conference Softmoderne - Literatur im Netz*. 1998.

- Essl, Karlheinz. « Lexikon-Sonate ». In *ton-gemisch. darmstadt-lectures*. Lothar Knessler, 1994.
- . « Lexikon-Sonate : An Interactive Realtime Composition for Computer-Controlled Piano ». In *Communications of the ICMA*, t. 16. 1996.
- . RTC-lib 5.0, for Mac OSX, Max 5, 2010.

Sources secondaires

- Adorno, Theodor W. *Ästhetische Theorie*. Suhrkamp Verlag KG, 1970.
- . *Philosophie der neuen Musik*. Gallimard, 1949.
- Appleton, Jon, Curtis Roads et John Strawn. *Composers and the Computer*. JSTOR, 1986.
- Assayag, Gérard, et Andrew Gerzso. *New computational paradigms for computer music*. T. 17. Delatour, 2009.
- Bayle, Laurent. *Composition et environnements informatiques*. IRCAM-Centre Georges-Pompidou, 1992.
- Breton, Philippe. *Histoire de l'informatique*. La découverte Paris, 1987.
- Cauquelin, Anne. *Les théories de l'art : « Que sais-je ? » n. 3353*. Presses universitaires de France, 2010.
- Collins, Nick. *Introduction to computer music*. John Wiley & Sons, 2010.
- Collins, Nick, Alex McLean, Julian Rohrhuber et Adrian Ward. « Live coding in laptop performance ». *Organised sound* 8, n° 03 (2003) : 321–330.
- Cope, David. *The algorithmic composer*. T. 16. AR Editions, Inc., 2000.
- Cortada, James W. *The Digital Hand, 3 vols*. T. 8. 2004.
- Couchot, Edmond, et Norbert Hillaire. *L'art numérique*. Flammarion, 2003.
- Dobretsberger, Christine. « Karlheinz Essl : Der Komponist als Zufallsgenerator ». In *Mozarts Erben*. Ibera Verlag, 2006.
- Donin, Nicolas, Laurent Feneyrou et Pierre-Laurent Aimard. *Théories de la composition musicale au XXe siècle*. T. 1. Symétrie, 2013.
- Eckel, Gerhard. « About the Installation of Karlheinz Essl's Lexikon-Sonate ». 1995.
- . « Technological Musical Artifacts ».
- Eco, Umberto. *L'Œuvre ouverte*. Points, 1962.
- Ehrler, Hanno. « Der Wiener Komponist Karlheinz Essl ». 1999.
- Essl, Karlheinz, et Annelies Kühnelt. « Rückblick / Vorschau ». *AKKORD (Zeitschrift der Musikschule Klosterneuburg)* 1 (2006/2007).
- Felber, Andreas. « Der verlängerte Schreibtisch des Komponisten ». In *PASSION FOR ART, Ausstellungskatalog*. Sammlung Essl Privatstiftung, 2007.

- Förster, Jonas. « Intuition, Automation und Entscheidung. Der Komponist im Prozess algorithmischer Komposition ». Mémoire de maîtrise, Folkwang Universität der Künste Essen, 2011.
- Goethe, Johann Wolfgang, W. Nöldeke et Raimund Oehler. *Italienische Reise*. Reclam, 1817.
- Günther, Bernhard. « Irreal-Enzyklopädie – EINER METAPHORISCHEN REISE – ZUR Lexikon-Sonate VON Karlheinz ESSL ». 1995.
- Harvey, Brian. Computer science Logo style, Vols. 1–3, 1985.
- Heinrich, Marie-Noël. *Création musicale et technologies nouvelles : Mutation des instruments et des relations*. Editions L'Harmattan, 2003.
- Hiller, Lejaren Arthur, et Leonard M Isaacson. *Experimental Music ; Composition with an electronic computer*. Greenwood Publishing Group Inc., 1979.
- Jamie, James. *La Musique des sphères*. Editions du Rocher, 1997.
- Klein, Julieanne. « A Portrait of the Composer Karlheinz Essl ». *Fowl Feathered Review* 4 (2013) : 74–81.
- Knuth, Donald Ervin. *The art of computer programming, 4 vols.* T. 3. Pearson Education, 1997–2011.
- Koenig, Gottfried Michael. « Kommentar. Zu Stockhausen : ... wie die Zeit vergeht... Zu Fokker : Wozu und Warum ? Und zur augenblicklichen Praxis aus der Sicht des Autors ». 1962.
- Kuhn, Thomas S. *The Structure of Scientific Revolutions*. University of Chicago Press, 1970.
- Lacoste, Jean. *La philosophie de l'art : « Que sais-je ? » n. 1887*. Presses universitaires de France, 2010.
- Lippe, Cort, Zack Settel, Miller Puckette et Eric Lindemann. « The IRCAM Musical Workstation : A Prototyping and Production Tool for Real-Time Computer Music ». 1991.
- Manoury, Philippe, Omer Corlaix et Jean-Guillaume Lebrun. *La musique du temps réel : entretiens avec Omer Corlaix et Jean-Guillaume Lebrun*. Editions MF, 2012.
- Miller, Arthur I. *Colliding worlds : how cutting-edge science is redefining contemporary art*. WW Norton & Company, 2014.
- Moles, Abraham. « Art et ordinateur ». *Communication and languages* 7, n° 1 (1970) : 24–33.
- Moore, Gordon. « Cramming more components onto integrated circuits ». *Readings in computer architecture* 56 (2000).
- Neeman, Elsa, Jérôme Meizoz, Claire Clivaz et al. « Culture numérique et auctorialité : réflexions sur un bouleversement ». *A contrario*, n° 1 (2012) : 3–36.
- Okopenko, Andreas. *Lexikon-Roman einer sentimental Reise zum Exportertreffen in Duden*. 26086. Ullstein, 1983.

- Paloque-Berges, Camille, et Christophe Masutti. *Histoires et cultures du Libre. Des logiciels partagés aux licences échangées*. Lulu. com, 2013.
- Puckette, Miller. « Combining event and signal processing in the MAX graphical programming environment ». *Computer music journal* 15, n° 3 (1991) : 68–77.
- Reimer, Jeremy. « Total share : 30 years of personal computer market share figures ». *Ars Technica* 15 (2005).
- Rowe, Robert. *Machine musicianship*. MIT press, 2004.
- Schäfer, Georg E. *History of Computer Science : Technology, Application and Organization*. BoD–Books on Demand, 2013.
- Schechner, Richard, Marie Pecorari, Anne Cuisset et Christian Biet. *Performance : expérimentation et théorie du théâtre aux USA*. Éd. théâtrales, 2008.
- Scholl, Steffen. « Karlheinz Essls RTC-lib ». In *Musik – Raum – Technik. Zur Entwicklung und Anwendung der graphischen Programmierumgebung »Max«*, 102–107. Transcript Verlag, 2014.
- Sinkovicz, Wilhelm. « Fantasie als Sprengstoff ». *DIE PRESSE* (2005).
- Von Neumann, John. « 13. Various Techniques Used in Connection With Random Digits » (1951).
- Weberberger, Doris. « Porträt : Karlheinz Essl ». *MUSIC AUSTRIA* (2012).
- Wilson, Stephen. *Art+Science Now*. Thames & Hudson, 2012.
- Wittgenstein, Ludwig. *Philosophische Untersuchungen*. 1953.
- Wodon, Bernard. *Histoire de la musique*. Larousse, 2014.
- Xenakis, Iannis. « Musiques Formelles Nouveaux Principes Formels de Composition Musicale » (1981).