COLLOQUE SUR LES NOUVEAUX PARADIGMES
POUR L'INFORMATIQUE MUSICALE
**11 JUIN 2007**

**COLLOQUIUM ON NEW COMPUTATIONAL PARADIGMS FOR COMPUTER MUSIC**
**JUNE 11, 2007**

Organisateurs : Andrew Gerzso (directeur Médiations Recherche/Création, Ircam),
Gérard Assayag (responsable équipe Représentations musicales, Ircam)
*Organizers: Andrew Gerzso (Director of the Department of Creative and Scientific Exchanges, IRCAM),*
*Gérard Assayag (Head of Musical Representations, IRCAM)*

Coordination : Florence Quilliard
Traduction des textes : Deborah Lopatin
*Translation: Deborah Lopatin*

Entrée libre dans la limite des places disponibles
*Free entrance on a first-come, first-serve basis*

COLLOQUE SUR LES NOUVEAUX PARADIGMES
POUR L'INFORMATIQUE MUSICALE

IRCAM
SALLE IGOR-STRAVINSKY

LUNDI 11 JUIN
9H30 - 19H

## LUNDi 11 JUiN 2007

### ■ 9H30-10H
**Accueil**
**Gérard Assayag** (responsable équipe Représentations musicales, Ircam), **Andrew Gerzso** (directeur Médiations Recherche/Création, Ircam)

### ■ 10H-11H
**Les principaux paradigmes de programmation**
**Peter Van Roy** (université Catholique de Louvain, Belgique) - Conférence invitée

■ 11H-11H30 - PAUSE

### ■ 11H30-12H15
**L'approche par modèles formels des processus musicaux**
**Camilo Rueda** (Universidad Javeriana, Cali, Colombie)

### ■ 12H15-13H
**Interaction dans le calcul**
**Carlos Agon** (équipe Représentations musicales, Ircam, France)

■ 13H-14H30 - DÉJEUNER

### ■ 14H30-15H30
**De l'art combinatoire à la combinaison des arts**
**Philippe Codognet** (université Paris 6, France, et Ambassade de France au Japon) - Conférence invitée

### ■ 15H30-16H15
**Sur l'évolution historique des langages de programmation**
**Pierre Cointe** (Projet Obasco, EMN-INRIA et LINA-CNRS, France)

■ 16H15-16H45 - PAUSE

### ■ 16H45-17H30
**Le paradigme de la programmation créative**
**Yann Orlarey** (directeur scientifique, Grame, Lyon, France)

### ■ 17H30-18H15
**Temps et perception en musique et informatique (et vice-versa)**
**Alexandre R. J. François** (Viterbi School of Engineering, USC, USA)

### ■ 18H15-19H
**Discussion et synthèse**
**Gérard Assayag** - modérateur

## MONDAY JUNE 11, 2007

### ■ 9:30AM-10AM
***Welcome***
***Gérard Assayag*** *(Head of Musical Representations, IRCAM),*
***Andrew Gerzso*** *(Director of the Department of Creative and Scientific Exchanges, IRCAM)*

### ■ 10AM-11AM
***A Survey of the Main Programming Paradigms***
***Peter Van Roy*** *(Catholic University of Louvain, Belgium)*
*Keynote Speaker*

■ 11AM-11:30AM - BREAK

### ■ 11:30AM-12:15PM
***The Formal Models Approach to Musical Processes***
***Camilo Rueda*** *(Universidad Javeriana, Cali, Combia)*

### ■ 12:15PM-1PM
***Interaction in Computation***
***Carlos Agon*** *(Musical Representations team, IRCAM, France)*

■ 1PM-2:30PM - LUNCH

### ■ 2:30PM-3:30PM
***From the Art of Combinatorics to the Combination of Arts***
***Philippe Codognet*** *(University Paris 6, France, and French Embassy in Japan) - Keynote Speaker*

### ■ 3:30PM-4:15PM
***On the Evolution of Programming Languages***
***Pierre Cointe*** *(Obasco Project, EMN-INRIA and LINA-CNRS, France)*

■ 4:15PM-4:45PM - BREAK

### ■ 4:45PM-5:30PM
***The Paradigm of Creative Programming***
***Yann Orlarey*** *(Scientific Director, GRAME, Lyon, France)*

### ■ 5:30PM-6:15PM
***Time and Perception in Music and Computation (and Vice-Versa)***
***Alexandre R. J. François*** *(Viterbi School of Engineering, USC, USA)*

### ■ 6:15PM-7PM
***Discussion and Conclusions***
***Gérard Assayag*** *- Moderator*

# NOUVEAUX PARADIGMES POUR L'INFORMATIQUE MUSICALE
# NEW COMPUTATIONAL PARADIGMS FOR COMPUTER MUSIC

Aujourd'hui, nombre de technologies pour la musique - logiciel et matériel - sont basées sur des paradigmes qui étaient déjà présents quand le domaine de l'informatique musicale s'est créé, il y a trente ans. La recherche à l'Ircam, motivée par des objectifs musicaux précis, a souvent été à la pointe de la science informatique. Le développement des systèmes temps réel (4X, Max/MSP), les langages orientés objets (Formes), la programmation par contraintes (Situation) et visuelle (OpenMusic) en sont autant d'exemples.

*Much of today's computer music technology, both hardware and software, is based on computational paradigms that were present when the field emerged almost thirty years ago. IRCAM's research, motivated by specific musical goals, has frequently been on the cutting edge of computer science. The development of real-time systems (e.g. 4X, Max/MSP), object oriented languages (Formes), constraint programming (Situation), and visual programming languages (OpenMusic) are some examples.*

Aujourd'hui, l'évolution du développement technologique pour la musique a atteint un niveau de maturité conceptuelle et technique tel que, lorsqu'on regarde vers l'avenir, il est vital d'examiner les nouveaux paradigmes de calcul qui ont émergé dans la science informatique afin d'évaluer leur potentiel et leur pertinence pour les applications futures. Ces idées pourraient, d'un côté, servir de source d'inspiration pour des projets musicaux et, de l'autre, être utilisées pour résoudre des problèmes jusqu'ici sans solution. En même temps, certains problèmes rencontrés dans la musique informatique – comme, par exemple, la représentation du temps sous ses différentes formes – pourraient être une source d'intérêt pour la communauté de la science informatique.

*Today, many of computer music's major developments have reached a certain conceptual and technical maturity. As a result, it is vital to examine the new computational paradigms that have emerged in computer science in order to evaluate their relevance and potential for the applications for music in the years ahead. On one hand, the new ideas could serve as a source of inspiration for musical projects, and on the other they could serve as solutions for known – but up to now unsolved – problems. At the same time, some problems encountered in computer music – such as the representation of time in its different musical forms – could be a source of interest to the computer-science community.*

Le but de ce colloque est ainsi de rassembler des acteurs majeurs aussi bien dans le domaine de l'informatique scientifique que musicale afin de susciter des échanges à plusieurs niveaux et esquisser quelques perspectives pour le futur.

*The colloquium's goal, therefore, is to bring together major figures in the fields of both computer science and music in order to encourage exchanges on different levels and outline new perspectives for the future.*

**Gérard Assayag**
**Andrew Gerzso**

## ■ 9H30-10H

### ACCUEIL
-

**Gérard Assayag** (responsable équipe Représentations musicales, Ircam), **Andrew Gerzso** (directeur Médiations Recherche/Création, Ircam)

## ■ 9:30AM-10AM

### *WELCOME*
-

***Gérard Assayag*** *(Head of Musical Representations, IRCAM)*
***Andrew Gerzso*** *(Director of the Department of Creative and Scientific Exchanges, IRCAM)*

## ■ 10H-11H

### LES PRINCIPAUX PARADIGMES DE PROGRAMMATION
-

**Peter Van Roy** (université Catholique de Louvain, Belgique) - Conférence invitée

La programmation est une discipline riche. Il y a beaucoup de manières différentes de programmer un ordinateur, appelées des paradigmes de programmation. Dans cet exposé, je mettrai un peu d'ordre dans les 20 paradigmes les plus importants. Je survolerai largement et profondément tous les concepts les plus importants de la programmation. Je présenterai d'abord les concepts les plus connus, dans la programmation déclarative et la programmation orientée objet et, ensuite, j'expliquerai les développements plus récents dans la programmation multi-agent. Il est important de réaliser que tous les paradigmes que nous verrons sont d'importance égale : un programme devrait utiliser celui qui est approprié et il y aura souvent plusieurs paradigmes dans un même programme.
Le premier paradigme est la programmation déclarative, qui peut être faite avec des fonctions (qui sont directionnelles) ou des relations (qui sont non directionnelles). La programmation par contraintes est une forme de programmation relationnelle basée sur des algorithmes sophistiqués de résolution de certaines formes de relations. Dans la programmation fonctionnelle, un programme est défini avec des fonctions pures. Une fonction est « pure » quand elle donne toujours le même résultat avec les mêmes arguments : elle est déterministe, indépendante du reste du système et n'a pas de mémoire. La pureté est une propriété importante parce qu'un programme qui est correct à un moment donné restera correct pour toujours. Les fonctions sont des entités de première classe : elles peuvent être des arguments et des résultats d'autres fonctions. La programmation déclarative contient en forme embryonnaire beaucoup des idées se rapportant à d'autres paradigmes.
Le paradigme suivant est la programmation orientée objet. Elle étend la programmation déclarative avec l'abstraction de données, l'état explicite, le polymorphisme et l'héritage. L'abstraction de données permet le fractionnement des programmes en diverses parties appelées « abstractions » qui peuvent être développées indépen-

## ■ 10PM-11PM

### *A SURVEY OF THE MAIN PROGRAMMING PARADIGMS*
-

***Peter Van Roy*** *(Catholic University of Louvain, Belgium) - Keynote Speaker*

*Programming is a rich discipline. There are many different ways to program a computer, called programming paradigms. In this talk I will bring some order into more than 20 important paradigms. I go into a broad and in-depth survey of all the most important concepts in programming. I will first go over the more well-known concepts in declarative programming and object-oriented programming and then I will explain the more recent developments in multi-agent programming. It is important to realize that all of the paradigms that we will see are equally important: programs should use whichever is appropriate and there will often be more than one paradigm in a program.*
*The first paradigm is declarative programming, which can be done either with functions (directional) or relations (non-directional). Constraint programming is a form of relational programming based on sophisticated algorithms for solving certain kinds of relations. Functional programming consists of defining programs as pure functions. By "pure" we mean that a function always gives the same result when called with the same arguments: it is deterministic, independent of the rest of the system, and has no memory. Purity is an important property because it means that programs that are correct at one point in time will always be correct. Functions are first-class entities: they can be passed to other functions and returned as results. Declarative programming contained in embryonic form many of the ideas of later paradigms.*
*The next paradigm is object-oriented programming. It extends declarative programming with data abstraction, explicit state, polymorphism, and inheritance. Data abstraction allows programs to be partitioned into abstractions that can be developed independently. Explicit state adds memory to programs, which introduces a time dimension. Its most important use is modularity: an abstraction can be changed without changing the rest of the program. Polymorphism lets programs be structured according to responsibilities: a particular task can be completed by a single abstraction which is completely responsible for it. Inheritance lets the common parts of abstractions be factored out.*

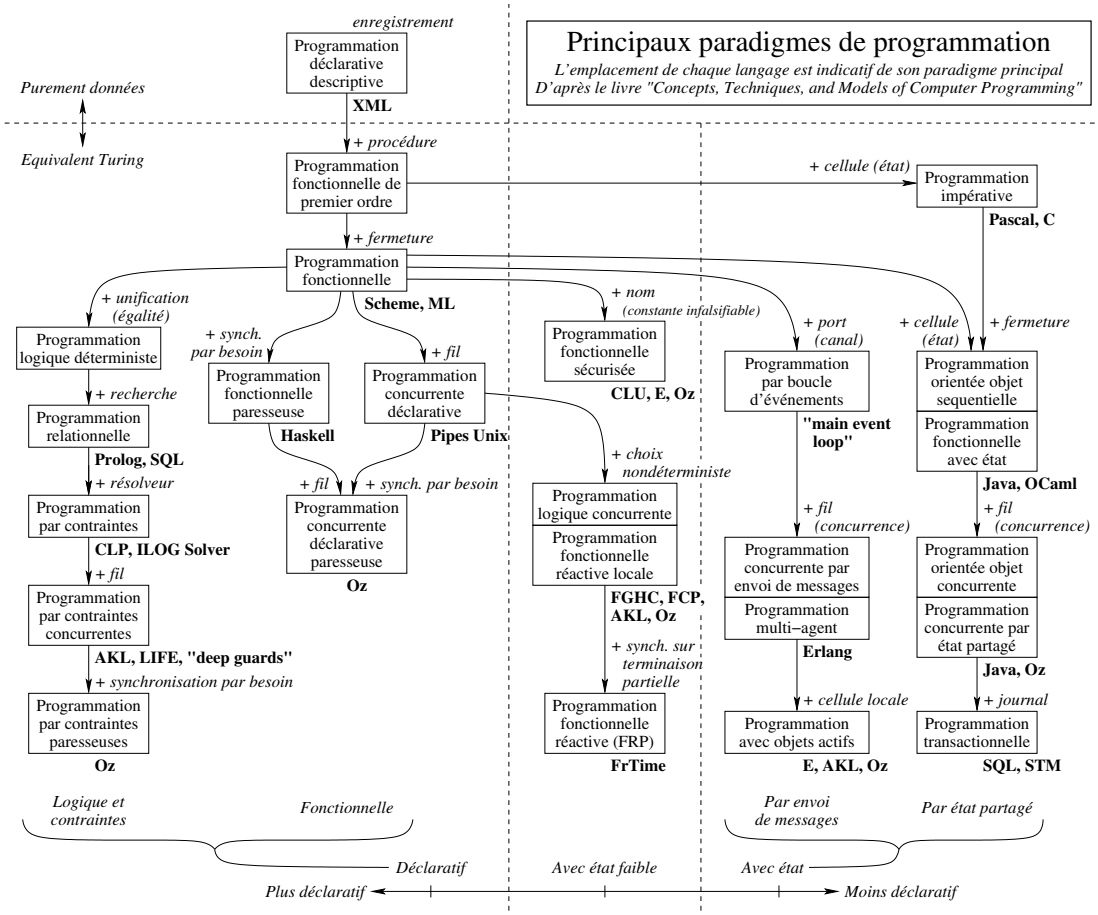
damment. L'état explicite ajoute de la mémoire aux programmes, ce qui introduit une dimension temporelle. Son utilisation la plus importante est pour la modularité : une abstraction peut être changée sans changer le reste du programme. Le polymorphisme permet de structurer les programmes selon les responsabilités : une tâche particulière peut être traîtée complètement par une seule abstraction, qui en est responsable. L'héritage permet d'isoler les parties communes des abstractions en un seul endroit du programme.

Le paradigme final est la programmation multi-agent, qui est moins bien compris que les autres mais pas moins important. Il est basé sur le constat que le monde est plein d'activités indépendantes et d'événements inattendus. Les activités indépendantes sont appelées « concurrentes ». Si en plus elles s'exécutent sur des ordinateurs différents, elles s'appellent « réparties ». Des événements inattendus, bénins ou malicieux, peuvent arriver à tout moment. La programmation multi-agent comporte des « agents » indépendants qui interagissent pour réaliser les buts globaux du programme. Il faut maîtriser le comportement des agents pour qu'ils collaborent, avec des opérations de base comme l'exclusion mutuelle, les barrières, les situations figées, les mises à jour cohérentes et différentes formes de protocoles de négociation. La concurrence peut être réalisée en trois manières principales : la première est la concurrence déclarative (qui est aussi pure que la programmation déclarative mais pas toujours assez expressive), ensuite vient la concurrence par envoi de messages (agents indépendants qui s'envoient des messages) et finalement la façon la plus difficile qui est la concurrence partagée (des agents qui partagent des données communes). Les contingences sont traitées par des transactions et des décisions temporisées prises au bon niveau d'abstraction, pas avec des exceptions et des dépassements de temps. Nous expliquons pourquoi ces derniers sont de mauvaises idées.

Un bon système de programmation devrait soutenir plusieurs paradigmes. Nous donnons des exemples dans le langage Oz, qui soutient la plupart des paradigmes et pour lequel il existe une bonne implémentation en logiciel libre, Mozart (voir www.mozart-oz.org). Nous recommandons aussi le livre « Concepts, Techniques, and Models of Computer Programming », qui présente la plupart des paradigmes dans un cadre uniforme qui montre leurs relations.

*The final paradigm is multi-agent programming, which is less well understood than the others but no less important. It is based on the realization that the world is full of independent activities and unexpected events. Activities that are independent are called concurrent. If, in addition, they run on different computers, they are then called distributed. Unexpected events, either benign or malicious, can happen at any time. Multi-agent programming consists of independent "agents" that interact with each other to achieve the overall goals of the program. Programming consists of harnessing the agents to work together, using primitives such as mutual exclusion, barriers, snapshots, checkpoints, and different forms of negotiation protocols. Concurrency can be realized in three principal ways: the first is declarative concurrency (which is just as pure as declarative programming but is not always expressive enough), then comes message-passing concurrency (independent agents sending messages to each other), and finally the hardest way is shared-state concurrency (agents sharing common data). Contingencies are handled with transactions and timed decisions taken at the right level of abstraction, not with exceptions and time outs. We will explain why the latter are bad ideas.*

*A good programming system should support multiple paradigms. We give examples using the Oz language, which supports most of the paradigms and for which there is a good Open Source implementation—Mozart (see www.mozart-oz.org). We also recommend the book* Concepts, Techniques, and Models of Computer Programming, *which presents most paradigms in a structured way showing their relationships.*




**Peter Van Roy** est membre du département d'Ingénierie Informatique à l'université catholique de Louvain, Louvain-la-Neuve, Belgique. Il est ingénieur de la Vrije Universiteit Brussel (1983) et il a obtenu un master en science et un doctorat en informatique de l'université de Californie, Berkeley (1984, 1990). Il a une habilitation à diriger des recherches de l'université Paris 7 (1996). Il a travaillé au Digital Paris Research Laboratory, au Deutsches Förschungszentrum für Künstliche Intelligenz et au Swedish Institute of Computer Science. Il est l'auteur de Aquarius Prolog, le premier système qui peut exécuter le langage logique Prolog avec la même efficacité que le langage C. Il est un implémenteur du langage LIFE, un successeur de Prolog, et un développeur du langage multi-paradigme Oz. Avec Seif Haridi et Per Brand, il a étendu Oz pour la programmation répartie, un des premiers systèmes qui montre que la répartition transparente est une approche pratique. Il a développé et breveté le logiciel de dessin FractaSketch, qui a été utilisé par une grande maison de couture pour hommes. Il est co-auteur avec Seif Haridi du livre « Concepts, Techniques, and Models of Computer Programming », MIT Press, 2004, qui est largement reconnu comme un nouveau classique du genre. Il travaille actuellement sur la programmation des systèmes auto-gérants (« self-managing systems »).

**Peter Van Roy** *is a member of the Computer Engineering Department at the Université Catholique de Louvain, in Louvain-la-Neuve, Belgium. Van Roy received his engineering diploma from the Vrije Universiteit Brussel in 1983 before going on to the University of California where he received his MS (1984) and PhD (1990). Peter Van Roy was accredited to supervise research at the Université Paris 7 in 1996. He has worked at Digital Paris Research Laboratory and at the Swedish Institute of Computer Science. He is the author of Aquarius Prolog, the first system that can carry out the Prolog language with the same level of efficiency as C. He implemented the LIFE language (the successor of Prolog) and developed the multi-paradigm language, Oz. He expanded Oz for distributed programming with Seif Haridi and Per Brand. This was one of the first systems that demonstrated that transparent distribution is a practical approach. Van Roy also developed and patented the drawing software program, FractaSketch that was used by an important menswear design group. In 2004, he published the book* Concepts, Techniques, and Models of Computer Programming *(MIT Press) that he wrote in collaboration with Seif Haridi. This book is largely recognized as a new classic in its genre. Peter Van Roy is currently working on programming self-managing systems.*

■ 11H-11H30 - PAUSE

## ■ 11H30-12H15

## L'APPROCHE PAR MODÈLES FORMELS DES PROCESSUS MUSICAUX

-

**Camilo Rueda** (Université Javeriana, Cali, Combie)

Le calcul de processus concurrents s'est établi depuis les années 90 pour modéliser des systèmes dans un grand nombre de domaines. L'idée fondamentale dans ce type de calcul est de remplacer la notion de *fonction* par celle de *processus* dans la conception qu'on se fait d'un calculateur. Plus précisément, il s'agit de remplacer la notion de lecture/écriture d'un symbole dans une mémoire, vue comme le pas élémentaire de calcul, par celle d'exécution d'une *interaction*. Le but est naturellement d'atteindre une formalisation plus naturelle du comportement des logiciels modernes, interactifs et distribués, que l'on a du mal à décrire de manière purement fonctionnelle.
Deux chemins ont été explorés pour le calcul de processus :
• Les modèles de processus concurrents basés sur le pi-calcul de Milner ;
• Les modèles de contraintes concurrentes basés sur le calcul *ccp* de Saraswat.
Dans le pi-calcul, les processus P et Q interagissent en envoyant ou en recevant de l'information sur un *canal* de communication commun. Dans *ccp*, les agents P et Q interagissent en ajoutant ou en consultant une information dans une collection commune de variables. Une différence centrale dans le cas de *ccp* est que les informations relatives aux variables peuvent être *partielles*, dans le sens où l'information relative à une variable n'a pas besoin d'être suffisante pour lui donner une valeur précise. Par exemple, on pourrait formuler la contrainte que la hauteur d'une note appartient à un ensemble donné de valeurs MIDI, sans pour autant spécifier sa valeur exacte.
Il y a au moins trois raisons de penser que le calcul de processus, d'une sorte ou d'une autre, puisse faire ses preuves dans la modélisation des processus musicaux interactifs — en particulier dans le cas de la performance musicale ou de l'improvisation. D'abord, ces formalismes sont *compositionnels*. C'est-à-dire qu'un système peut être modélisé à un certain niveau d'abstraction de manière à comprendre précisément son comportement global, puis une hiérarchie de modèles peut ensuite raffiner la description, chacun ajoutant des observations de plus en plus détaillées. Les relations exactes entre tous les modèles de la hiérarchie sont dérivées de manière précise de la sémantique du calcul. Deuxièmement, les propriétés désirables (resp. indésirables) du système peuvent être formellement démontrées (resp. invalidées) à l'intérieur du modèle avant même de procéder à une simulation. Troisièmement, le modèle de calcul est *computationnel* dans le sens où il peut être directement interprété comme un programme informatique qui définit alors un simulateur du système.

■ 11AM-11:30AM - BREAK

## ■ 11:30AM-12:15PM

## *THE FORMAL MODELS APPROACH TO MUSICAL PROCESSES*

-

***Camilo Rueda*** *(University Javeriana, Cali, Combia)*

*Concurrent process calculi have been used since the 90s for modeling systems in a wide variety of areas. The strong idea in these calculi is to replace the fundamental notion of a computing device from that of* function *to that of* process *or, more precisely, to replace the basic computation step from that of reading or writing a symbol in a tape to that of performing an* interaction*. The whole point is, of course, to have a more natural way of formally expressing the behavior of modern interactive software (e.g. a distributed system) which can hardly be described functionally.*
*Two main themes have been explored for process calculi:*
*• Concurrent processes models based on Milner's pi-calculus*
*• Concurrent constraints models based on Saraswat's* ccp *calculus*
*In the pi-calculus concurrent processes P and Q interact by sending or retrieving information along a common* channel*. In* ccp *agents P and Q interact by adding or reading information in a collection of common variables. A central difference is that in* ccp*, information on variables can be* partial*, in the sense that the information available on a variable does not have to be enough to determine a precise value for it. For example, one might wish that the pitch of a note belonging to some given set of MIDI's, without having to specify its exact value.*
*There are at least three reasons why process calculi of one or the other variety could prove to be especially valuable in modeling systems of interacting musical processes — particularly in the context of music performance/improvisation. First, those formalisms are* compositional*. That is, a system can be modeled at some level of abstraction in order to understand precisely its global behavior then refined in a hierarchy of models each adding more and more detailed observations. The exact relationship between all models in the hierarchy is derived in a precise way from the semantics of the calculus. Second, (un)desirable properties of the system can be formally (dis)proved within the model before launching any simulation. Third, the calculus model is* computational *in the sense that it can be directly interpreted as a computer program defining a simulator of the system.*
*In many musical settings material is implicitly defined by rules. In most cases these rules do not give the whole account of the actual musical material the composer has in mind but provide only rough information on some of its aspects. Any model of this process would then have to define the evolution of the musical system based only on partial information. For this reason the* ccp *approach seems particularly adapted.*
*Music, however, poses difficult challenges that question the relevance of these models. One is the notion of time. The* ccp *calculus has been extended to handle a linear, non-metric,*

Dans beaucoup de situations musicales, le matériau est implicitement défini par des règles. Dans la plupart des cas, ces règles ne rendent pas compte de la totalité du matériau que le compositeur a en tête, mais fournissent seulement des informations grossières sur certains de ses aspects. Toute modélisation de ce processus devrait définir l'évolution du système musical en se basant uniquement sur des informations partielles. Pour cette raison, l'approche *ccp* semble particulièrement adaptée.
La musique cependant pose des défis difficiles qui questionnent la pertinence de ces modèles. Un de ces défis est relatif à la notion de temps. Le calcul *ccp* a été étendu de manière à incorporer un modèle de temps linéaire, non métrique et discret. Dans le formalisme *ntcc*, par exemple, cette notion simple permet de définir une logique temporelle associée au calcul de sorte que les propriétés d'un processus, comme l'exécution éventuelle d'un accord à l'intérieur d'un intervalle temporel, peuvent être formulées de manière pratique et prouvées automatiquement. Deux questions centrales subsistent cependant. Comment intégrer une métrique dans cette structure temporelle simple, de manière à mieux prendre en compte les propriétés rythmiques sans rien perdre de la cohérence logique du calcul. Et comment intégrer des facilités temps-réel comme l'interruption asynchrone des processus, question particulièrement sensible dans la performance et l'improvisation.
Une autre question cruciale est celle du *choix* et du non-déterminisme. En musique, on peut identifier des dimensions horizontales et verticales du non-déterminisme. La dimension horizontale est relative à l'éventualité temporelle. Une note peut être planifiée pour débuter à une date indéterminée au sein d'un intervalle temporel. L'autre dimension est relative aux choix indéterministes entre plusieurs chemins d'évolution du système. Ces deux aspects sont proprement traités dans *ntcc*. Dans plusieurs situations musicales, cependant, les *choix* sont stochastiques plutôt qu'indéterministes. Le défi est alors de savoir comment intégrer des décisions probabilistes dans un calcul *ccp* tout en préservant la possibilité de prouver logiquement les propriétés des processus dans un cadre logique adapté.
Dans cette présentation, ces thèmes seront explorés en détail et des lignes possibles de développement proposées.

*and discrete model of time. In the* ntcc *formalism, for example, this simple notion makes it possible to define a temporal logic associated to the calculus so that properties of a process, such as the eventual output of some chord within some given time interval, can be conveniently formulated and automatically proved. Two central questions remain to be answered. One is how to integrate a metric in this simple notion of time and so to better handle rhythmic properties in such a way that the logical counterpart of the calculus is not lost. The other, pertinent to performance and improvisation, is how to include real-time aspects such as time interruptible processes.*
*Another central question is the notion of* choice*. Related to choice is non-determinism. In music one can identify a vertical and horizontal dimension of non-determinism. The horizontal dimension pertains to temporal eventuality. A particular note can be scheduled to sound at some undetermined point within a given temporal interval. The other dimension relates to an undetermined choice between two possible paths of evolution. These two aspects are neatly handled in* ntcc*. In many musical settings, however, instead of being non deterministic,* choices *are defined to be stochastic. The challenge here is, again, how to integrate probabilistic* choices *in a* ccp *based calculus such a way that properties of processes can still be proved using a suitable logic.*
*In this talk these issues will be explored in further detail and possible lines of development proposed.*

**Camilo Rueda** est directeur du département de sciences et génie de l'informatique à l'université Javeri ana-Cali. Il est directeur de l'équipe de recherche AVISPA composée de chercheurs de l'université Javeriana-Cali et de l'université del Valle en Colombie, de l'Ircam et du labo Lyx de l'Ecole Polytechnique à Paris, et du Cork Constraint Computation Centre en Irlande. Ses travaux de recherche portent sur la théorie et les applications de la programmation concurrente de contraintes et la modélisation formelle de systèmes interactifs par les calculs de processus concurrents. Il a participé au développement de plusieurs logiciels issus de ces recherches dans les domaines de la biologie, la sécurité informatique et la musique.

***Camilo Rueda*** *is the director of the Department of Computer Science and Engineering at the Universidad Javeriana-Cali. He is also the director of the AVISPA research group made up of scientists from IRCAM, the Lyx laboratory at the Ecole Polytechnique in Paris, and the Cork Constraint Computation Centre in Ireland. Camilo Rueda's work centers on the theory and application of concurrent and constraint programming and the formal modeling of interactive systems through concurrent computing processes. He has taken part in the development of several software programs as the result of his research in the domains of biology, computer security, and music.*

■ 12H15-13H

## INTERACTION DANS LE CALCUL
-

**Carlos Agon** (équipe Représentations musicales, Ircam, France)

Dans cette conférence, nous allons nous aventurer dans la « science-fiction » des rapports informatique/musique. Nous centrerons nos intuitions autour de la notion d'interaction. De manière générale, un système interactif se caractérise par la prise en compte, à tout moment, des informations fournies par l'utilisateur. Comme conséquence directe, les différentes entrées de l'utilisateur peuvent dépendre éventuellement des sorties du système. Cette manière de concevoir l'utilisation de l'ordinateur intègre l'utilisateur dans la conception, implémentation et évaluation d'un système informatique. Dans le cadre de l'informatique musicale et plus précisément dans la composition assistée par ordinateur (CAO), la prise en compte du compositeur pour l'élaboration d'un outil est l'une des différences principales avec la composition algorithmique.
Un système interactif est constitué par l'utilisateur, une interface et un noyau de calcul. Dans le cadre de la CAO, le dialogue entre utilisateur et calcul a pour but de lier deux composantes fondamentales dans la composition, à savoir, le raisonnement et le calcul lui-même. Dans le passé, nous avons réalisé plusieurs prototypes afin de pousser à l'extrême le rôle du compositeur dans son système de CAO. Avec l'introduction des langages de programmation (OpenMusic, par exemple) le compositeur a abandonné son rôle exclusif d'utilisateur d'une application spécifique pour devenir le programmeur de ses propres applications. D'autres tentatives ont été réalisées pour permettre au compositeur de devenir le concepteur de son propre langage. Concrètement, dans le cadre de la programmation par objets nous avons mis à portée de mains du compositeur une interface graphique du MOP du langage.

Ces différentes expériences nous ont permis de tracer un chemin vers le dialogue entre calcul et raisonnement, cependant la « route est longue ». Jusqu'à présent, dans les systèmes de CAO, le calcul reste une « boîte noire » pour le compositeur. Il est impossible pour un compositeur de modifier les règles de calcul au milieu d'une exécution. Ainsi, son point de vue sur les objets qu'il manipule reste une fois pour toutes défini par le calcul. On pourrait voir ces objets comme des vecteurs dans un espace vectoriel donné, le compositeur pouvant les modifier ou les combiner entre eux, mais sans jamais pouvoir modifier l'espace lui-même. Pour une interaction plus intéressante, il faudra définir ces objets de manière indépendante de la base, mais aussi introduire cette dernière dans le système même, de manière à pouvoir la modifier.

■ *12:15PM-1PM*

## *INTERACTION IN COMPUTATION*
-

***Carlos Agon*** *(Musical Representations team, IRCAM, France)*

*During this conference, we will journey into the "science fiction" of the computer/music relationship. We will center our intuitions on the notion of interaction. In general, an interactive system is characterized by the constant understanding of the information provided by the user. In direct consequence, the different entries by the user could depend on the results given by the system. This manner of imagining the use of a computer includes the user in the design, implementation, and testing of a computerized system. In the case of computer music, and more specifically in the case of computer-assisted composition, the acknowledgment of the composer for the creation of a tool is one of the major differences with algorithmic composition.*

*An interactive system is made up of a user, an interface, and a calculation kernel. In the case of computer-assisted composition, the dialogue between the user and the kernel combines two components that are fundamental in composition: the argument and the calculation itself. In the past, we created several prototypes in order to push the role of the composer to the limit in his computer-assisted composition system. With the introduction of programming languages such as OpenMusic, the composer has given up his role as exclusive user of a specific application to become the programmer of his own programs. Other attempts have been carried out so that the composer can design his own language. Concretely, we have placed the language's MOP graphical interface in the hands of the composer for object-oriented programming.*

*These different experiments have let us forge a path toward a dialogue between calculation and argument. However, the path will be long. Until today the calculation has been the "black box" of computer-assisted composition systems for the composer. It is impossible for the composer to modify the calculation rules in the middle of an action. Thus, his vision of the objects he is manipulating is defined by the calculation. These objects can be seen as vectors in a given vectoral space, the composer can modify or combine them but without ever modifying the space itself. For a more interesting interaction, it is necessary to define these objects independently from the beginning and introduce this notion into the system itself so that it can be altered.*
*The relationship between calculation and argument is, in my mind, the foundation of computer-assisted composition. In this talk, we will analyze the musical pertinence of different studies in computer science and in logic and how they could result in new forms of interaction between the composer and his computational system.*

Les rapports entre calcul et raisonnement sont, à notre sens, à la base de la CAO. Dans cette conférence, nous allons analyser la pertinence musicale des divers travaux en informatique et en logique et la façon dont ils pourront donner lieu à des nouvelles formes d'interaction entre le compositeur et son système computationel.

**Carlos Agon** est ingénieur en informatique. Son activité est reliée notamment à l'informatique musicale et plus particulièrement à la composition assistée par ordinateur (CAO). Il a développé diverses applications basées sur des techniques informatiques, comme la programmation visuelle, par contraintes et par objets. Actuellement, il est chargé de recherche à l'Ircam-CNRS (UMR 9912).

***Carlos Agon*** *is a computer science engineer. His work is linked to computer music and more specifically to computer-assisted composition. He has developed several applications based on computer techniques, like visual programming, using constraint programming and object-oriented programming. He is currently a researcher at IRCAM-CNRS (UMR 9912).*

■ 13H-14H30 - DÉJEUNER

■ *1PM-2:30PM - LUNCH*

## ■ 14H30-15H30

### DE L'ART COMBINATOIRE À LA COMBINAISON DES ARTS
-

**Philippe Codognet** (université Paris 6, France, et Ambassade de France au Japon) - Conférence invitée

Leibniz distinguait trois branches dans le champ du savoir : la logique, les arts de la mémoire (mnémonique) et l'art d'inventer (« ars inventendi », c'est à dire en fait la combinatoire), qui permettrait, par son exhaustivité, de « penser l'impensable» et d'ouvrir le processus créatif... Dans le champ de l'intelligence artificielle, l 'exploration exhaustive d'espace de possibilités a été un champ de recherche important depuis la fin des années 60. Il y a environ une vingtaine d'années, de nouvelles méthodes de calcul et le paradigme de la « programmation par contraintes » ont permis la mise en œuvre efficace d'algorithmes de recherche combinatoire dans les domaines discrets qui ont connu très vite des applications en composition musicale assistée par ordinateur (énumération dans un ensemble fini de possibilités, par exemple celui des notes ou des accords possibles). Plus récemment, les techniques dites de « recherche locale » ont aussi donné lieu à des applications musicales, par exemple le système OMCloud développé par Charlotte Truchet. L'une des caractéristiques essentielles de ce type d'algorithme est le processus de calcul « anytime », qui converge petit à petit vers une solution, sautant d'une solution partielle (optimum local) à une meilleure solution, et qui peut être arrêté à chaque instant. Cette caractéristique peut être utilisée par le compositeur comme un outil créatif, exhibant ainsi le processus créatif comme partie de l'œuvre. Cela a d'ailleurs été réalisé pour certaines œuvres musicales.
Est-ce donc un retour à Leibniz ?
Peut-on ainsi utiliser la perspective combinatoire et peut-être ce type d'algorithmes pour exhiber « la figuration d'un possible » (Marcel Duchamp) ?

## ■ *2:30PM-3:30PM*

### *FROM THE ART OF COMBINATORICS TO THE COMBINATION OF ARTS*
-

***Philippe Codognet*** *(University Paris 6, France, and French Embassy in Japan) - Keynote Speaker*

*G. W. Leibniz distinguished three branches in the field of knowledge: the art of logic, the art of memory (mnemonics), and the art of invention ("ars inventendi", that we now call combinatorics) , which makes it possible, through its exhaustiveness, to « think the unthinkable » and therefore feed the creative process.*
*In the field of Artificial Intelligence, exhaustive search algorithms have been an important area of research since the 60s. About twenty years ago, new computation techniques and the paradigm of "constraint programming" amount to efficient algorithms for combinatorial searches in discreet domains, which have been quickly applied to computer-based composition (for instance the exhaustive search within a finite set of possibilities, such as notes or chords). More recently, so-called "local search techniques" have also been applied to computer-based music (e.g. in the OMClouds sytem developed by Charlotte Truchet). One of the essential characteristics of this type of algorithm is the "anytime" computation process, which is converging little by little towards a global solution, hopping from one partial solution (local optimum) to a better one, and which can be stopped at anytime. This could be used by the composer as a tool, and thus exhibits the creative process as part of the artwork. This has indeed been achieved in some musical compositions.*
*Are we therefore going back to Leibniz's idea?*
*Can we use the combinatorics perspective, and maybe this kind of algorithm, to show "the figuration of a possible" (Marcel Duchamp)?*

Cette problématique s'inscrit donc plus généralement dans un ensemble de pratiques artistiques contemporaines, en particulier dans le champ des arts plastiques. Il est donc important de mettre le champ musical en perspective en s'interrogeant sur diverses œuvres liant art et nouvelles technologies qui font apparaître des processus similaires.

*These questions should be investigated more generally within the field of contemporary art practices, in particular in the field of visual arts. We will thus have to put this topic in perspective and link it to several artworks using new technologies which put similar processes at work.*

**Philippe Codognet** est actuellement Attaché pour la science et la technologie à l'Ambassade de France à Tokyo, en détachement de l'université de Paris 6 (Pierre et Marie Curie) où il est professeur.
Après des études de mathématiques et d'informatique à l'université de Bordeaux, il a obtenu un doctorat en informatique de cette université en 1989 et a rejoint l'INRIA (Institut National de Recherche en Informatique et l'Informatique) comme chercheur en 1990.
Après une année sabbatique au SONY Computer Science Laboratory à Paris en 1997/8, il devient professeur à l'université Pierre et Marie Curie (Paris 6) en septembre 1998. Ses recherches ont porté sur les langages de programmation, l'intelligence artificielle, la logique, les systèmes multi-agents et la réalité virtuelle. Il a participé à plusieurs projets nationaux et européens et plus de 70 publications dans des conférences et revues internationales décrivant en détail ses recherches.
En plus de ces activités purement scientifiques, il a travaillé depuis une dizaine d'années sur les relations entre l'art et les nouvelles technologies comme théoricien et critique d'art. Il a publié des essais dans des catalogues d'expositions ou des revues internationales et organisé des colloques interdisciplinaires sur les relations entre arts numériques, nouvelles technologies, art contemporain et performances multimédia. Il a été membre du conseil scientifique pour la recherche et l'innovation de la DAP (délégation aux arts plastiques) du ministère de la Culture de 2000 à 2003.

***Philippe Codognet** is currently Attaché for Science and Technology at the French Embassy in Japan (Tokyo), on leave from University of Paris 6 where he holds the position of professor in computer science.*
*Following studies in mathematics and computer science at the University of Bordeaux, he received a Ph.D. from this university in 1989 and then joined the French National Institute for Computer Science (INRIA) as senior researcher in 1990.*
*After a sabbatical leave from Sony Computer Science Laboratory in Paris in 1997/8, he joined University Pierre et Marie Curie (Paris 6) in September 1998 as full professor. His research focused on programming languages, artificial intelligence, logic, multi-agent systems, and virtual reality. He participated in many national and European projects, and over 70 publications in international journals and conferences detail his research. In addition to his purely scientific activities, he worked on the relationships between art and new technologies as theoretician and critic. He published articles in exhibition catalogues and international journals and organized several interdisciplinary symposiums on the relationships between digital arts, computer science, contemporary art, and multimedia performance.*
*He was member of the Scientific Council for research and innovation of the Visual Arts Department of the French Ministry of Culture from 2000 to 2003.*

## ■ 15H30-16H15

## SUR L'ÉVOLUTION HISTORIQUE DES LANGAGES DE PROGRAMMATION
-

**Pierre Cointe** (Projet Obasco, EMN-INRIA et LINA-CNRS, France)

« L'œuvre d'art est un *message* fondamentalement *ambigu*, une pluralité de signifiés qui coexistent en un seul signifiant (…).
Pour réaliser l'ambiguïté comme valeur, les artistes contemporains ont souvent recours à l'informel, au désordre, à l'indétermination des résultats. On est ainsi tenté d'établir une dialectique entre forme et ouverture, qui déterminerait dans quelle limite une œuvre d'art peut accentuer son ambiguïté et dépendre de l'intervention active du spectateur sans perdre pour autant sa qualité d'œuvre (U. Eco, *L'œuvre ouverte*). »

## ■ *3:30PM-4:15PM*

## *ON THE EVOLUTION OF PROGRAMMING LANGUAGES*
-

***Pierre Cointe** (Obasco Project, EMN-INRIA and LINA-CNRS, France)*

*-Please refer to the quote in French-*

Nous analysons l'évolution des langages de programmation en privilégiant deux points de vue. Celui de la dialectique entre « forme et ouverture » traduite par la définition de langages réflexifs supports à la construction d'architectures logicielles ouvertes. Celui de la tension entre généralité et spécialité telle qu'elle se manifeste avec l'apparition de langages métiers, dédiés exclusivement à un domaine, par opposition aux langages généralistes. Nous discutons finalement de l'apport potentiel des nouveaux paradigmes de programmation induits au domaine de l'informatique musicale.

*We analyse the evolution of programming languages by focusing on two concerns. On the one hand, we study the dialectic between form and opening leading to the definition of reflective languages to deal with open-ended software architectures. On the other hand, we observe the tension between the general and the specific as manifested by the development of domain-specific languages versus general-purpose languages. Finally, we discuss the impact of the associated programming paradigms on the field of musical computation.*

**Pierre Cointe** est un élève de Patrick Greussay et Jean-François Perrot. Sous leur direction, il soutient sa thèse d'Etat en 1984 à l'université Paris 6.
Il étudie d'abord la programmation fonctionnelle (implémentation d'un interprète ALisp en 1979) pour investiguer ensuite le domaine de la programmation par objets : réalisation en Lisp d'interprètes des langages d'acteurs Plasma et de classes Smalltalk (1980-1981). Il rejoint alors l'équipe Chant de Xavier Rodet à l'Ircam pour travailler à l'élaboration du langage de synthèse musicale Formes (1981-1986). De 1986 à 2002, au centre mondial de l'informatique puis chez Rank Xerox, à l'école des mines de Nantes et avec OTI (Object Technology International), il contribue au développement du principe des architectures réflexives en proposant des protocoles de méta-objets pour les langages Lisp (ObjVlisp), Smalltalk (ClassTalk), Self et Java. Depuis 2002, il s'intéresse à la programmation par aspects, sujet au cœur des recherches du projet INRIA OBASCO.

***Pierre Cointe** is a former student of Patrick Greussay and Jean-François Perrot. Under their direction he defended his "these d'Etat" at the University of Paris 6 in 1984.*
*He studied functional programming (implementation of an ALisp interpreter in 1979) before investigating the new area of object-oriented programming (implementation of Plasma and Smalltalk interpreters from 1980 to 1981). He went on to join the Chant team of Xavier Rodet at IRCAM to model and implement the Formes musical language (1981-1966). From 1986 to 2002, first at the Centre Mondial de l'Informatique and later at Rank Xerox , Ecole des mines of Nantes and with OTI (Object Technology International), he contributed to promote reflexive architectures by developing meta-object protocols for the Lisp (ObjVlisp), Smalltalk (ClassTalk), Self, and Java languages. Since 2002, his interests center around aspect-oriented programming, which is a core research subject of the INRIA OBASCO team.*

■ 16H15-16H45 - PAUSE

■ *4:15PM-4:45PM - BREAK*

## ■ 16H45-17H30

## LE PARADIGME DE LA PROGRAMMATION CRÉATIVE
-

**Yann Orlarey** (directeur scientifique, Grame, Lyon, France)

La programmation informatique classique vise à instruire un ordinateur des « choses » que l'on veut lui « faire faire ». Elle s'inscrit typiquement dans une démarche d'ingénieur, de type résolution de problème. L'objectif, le programme à écrire, le problème à résoudre, doit être défini et formalisé aussi clairement que possible avant même que le travail de programmation ne commence. Celui-ci se situe en bout de chaîne, après des phases d'analyse considérées comme plus nobles, et est confié à de simples programmeurs.
La « programmation créative », telle que nous l'aborderons dans cet exposé, est en rupture totale avec cette vision classique. Elle utilise la programmation informatique de manière radicale, comme support d'une démarche

## ■ *4:45PM-5:30PM*

## *THE PARADIGM OF CREATIVE PROGRAMMING*
-

***Yann Orlarey** (Scientific Director, GRAME, Lyon, France)*

*The goal of traditional computer programming is to teach a computer "things" that we want it to "do". It is usually a part of a long engineering process of the problem-solving type. The objective, the program to write, the problem to be solved should be defined and formalized as clearly as possible — even before the work of programming begins. This work is found at the end of the chain —following the phase of analysis that is considered nobler —and is given to simple programmers.*
*The "creative programming" that we will discuss during this conference is a complete break from this traditional point of view. It uses computer programming in a radical way, as a support for a processes of pure invention and creation. Therefore, in its most extreme definition the "thing" to "do"*

d'invention et de création pure. Par conséquent, dans sa définition la plus extrême, la « chose » à « faire faire » ne préexiste pas au travail de programmation, de la même manière que la musique à jouer ne préexiste pas au travail d'improvisation (même si cela se travaille, bien entendu). Elle se découvre et s'invente en la faisant, dans le processus d'interaction avec le langage de programmation, le programme et l'ordinateur. Elle se reconnaît comme étant la chose que l'on voulait faire au moment où soudain elle se produit.
En tant que paradigme d'usage particulier de la programmation, la « programmation créative » était en germe dès 1960 avec MUSIC III. Mais il est évident que la rapidité du cycle : modification du programme – obtention du résultat, est cruciale pour des pratiques actuelles comme le « live coding ». Elle l'est également dans le travail en studio. Il faut pour pouvoir programmer « à la vitesse des idées » avant qu'elles ne fuient. Ce sont là, bien entendu, des contraintes fortes, encore mal prises en compte par les langages de programmation et les modèles de calcul dominants, mais qui contribuent et contribueront de manière positive à l'évolution des technologies de programmation.

*does not exist before the work of programming begins, in the same way that music to be played does not exist before the work of performing begins (even if this is worked on before, of course). It discovers and invents itself by doing, in the process of interaction with the programming language, the program, and the computer. It recognizes itself as the thing the moment it suddenly happens.*

*As a paradigm of a particular use of programming, "creative programming" was in its infancy in 1960 with MUSIC III. But it is obvious that the rapidity of the cycle modifying a program—achieving a result is crucial for today's practices such as "live coding". It is also found in studio work. It is essential to be able to program "at the speed of an idea" before they vanish. It is here, of course, that constraints, still not accounted for by programming languages, but contribute in a positive way to the progress of programming technology.*

Parallèlement à des études universitaires en sciences économiques et en informatique, **Yann Orlarey** a suivi la classe de musique électroacoustique du conservatoire de Saint-Étienne. Membre du GRAME depuis 1983, il est actuellement directeur scientifique de cet organisme. Ses travaux de recherche portent principalement sur les langages formels et les systèmes d'exploitation temps-réel. Il est l'auteur ou le coauteur de différents systèmes et logiciels musicaux dont le système d'exploitation MidiShare et les langages MIDI Logo, MIDI Lisp, CLCE, Elody et Faust. Son répertoire musical comporte des pièces sur bande, des pièces interactives et des pièces instrumentales pour solistes, petites formations et orchestres. Pour la plupart, ses œuvres font appel à des moyens informatiques soit au niveau des situations de jeu instrumental proposées aux interprètes soit au niveau de la composition musicale proprement dite.

*In parallel to his university-level studies in economics and computer science, **Yann Orlarey** followed classes in electro-acoustic music at the conservatoire de Saint-Etienne. A member of GRAME since 1983, he is currently the scientific director of this organization.*
*Yan Orlarey's research centers principally on formal languages and real-time operating systems. He is the author, or co-author, of different musical systems and software programs including the MidiShare operating system and the MIDI Logo, MIDI Lisp, CLCE, Elody, and Faust languages. His musical repertoire includes works on magnetic tape, interactive works, and instrumental works for soloists, small groups, and orchestras. Generally, his works call on computer techniques in situations of instrumental playing offered to performers or for musical composition.*

## ■ 17H30-18H15

## TEMPS ET PERCEPTION EN MUSIQUE ET INFORMATIQUE (ET VICE-VERSA)
-

**Alexandre R. J. François** (Viterbi School of Engineering, USC, USA)

La musique est depuis longtemps un fascinant domaine d'application pour les informaticiens, un domaine particulièrement inspirant et stimulant, car il n'existe qu'à la confluence de la création, de la représentation et de la performance. Deux propriétés non « computationnelles » caractérisent les tâches musicales.

## ■ *5:30PM-6:15PM*

## *TIME AND PERCEPTION IN MUSIC AND COMPUTATION (AND VICE-VERSA)*
-

***Alexandre R. J. François** (Viterbi School of Engineering, USC, USA)*

*Music has long been a fascinating domain of application for computer scientists. It is a particularly challenging and inspiring one, as it only exists at the confluence of creation, representation, and performance. Two "un-computational" properties characterize these musical tasks. First, they are geared towards human perception. Computation aims at*

Premièrement, ces dernières sont dirigées vers la perception humaine. Le calcul vise lui, l'exactitude super-naturelle, l'ennuyeuse cohérence, et la reproductibilité absolue, alors que la perception serait le royaume de la variabilité naturelle, des attentes excitantes non satisfaites, et des reproductions approximatives. Deuxièmement, les tâches musicales sont liées au temps, d'une manière fondamentalement différente, forcées d'opter pour des compromis contradictoires dans la lutte constante entre le désir de stopper le temps et la nécessité de vivre l'expérience dans le présent.

Pour traiter la question posée par la modélisation computationnelle de tâches musicales, j'introduirai *Hermes/dl*,un nouveau langage de conception pour la spécification de systèmes dynamiques complexes. Comme langage, *Hermes/dl* est composé d'une collection de primitives, d'un ensemble de principes d'organisation, et de collections de situations qualifiantes. Une notation graphique protège l'utilisateur des traditionnelles notations algébriques, et rend *Hermes/dl* plus accessible et attrayant pour les utilisateurs potentiels des domaines artistiques et scientifiques. Une formalisation inscrite dans le cadre rigoureux de la théorie des graphes permet des manipulations mathématiques et sous-tend les outils de productivité et de collaboration qui assisteront l'utilisateur dans le processus de création. *Hermes/dl* implémente un nouveau modèle computationnel issu de mes recherches précédentes en architecture logicielle.

J'illustrerai l'utilisation des concepts développés dans *Hermes/dl* avec le design et la réalisation de *Mimi*, un système multi-modal interactif d'improvisation musicale qui explore le potentiel et l'impact puissant du feed-back visuel dans l'interaction musicien-machine. *Mimi* est un outil centré sur le musicien pour la performance et l'enseignement. Son composant clef et innovant est l'interface visuelle, conçue pour fournir au musicien une information instantanée et continue sur l'état du système. Pour l'improvisation humaine, dans laquelle le contexte et la planification sont primordiaux, l'état pertinent du système s'étend au passé récent et au futur proche. Le système *Mimi*, conçu et implémenté à l'aide du framework SAI (le précurseur de *Hermes/dl*) intègre avec succès le calcul symbolique et la synchronisation temps-réel dans un cadre d'interaction multi-modale. L'interface visuelle de *Mimi* permet un mélange particulier des réflexes immédiats courants dans l'improvisation et de la planification et du timing plus généralement associés avec la lecture musicale. *Mimi* n'est pas seulement un partenaire d'improvisation effectif ; il a aussi été validé comme plateforme permettant d'interroger les processus cognitifs impliqués dans l'improvisation.

Finalement, je bouclerai la boucle et défendrai la pertinence générale du modèle computationnel développé.

*Mimi* est un projet collaboratif avec Elaine CHEW (USC Viterbi School of engineering) et Dennis Thurmond (USC Thornton School of Music).

*super-natural exactness, boring consistency, and absolute reproducibility; on the other hand, perception is the realm of natural variability, exciting unmet expectations, and approximate live reproductions. Second, musical tasks relate to time in fundamentally different ways, forced to opt for opposite compromises in the constant struggle between the desire to stop time and the necessity to live (and experience) in the present. To address the challenges posed by the computational modeling of musical tasks, I will introduce* Hermes/dl, *a new design language for the specification of complex dynamic systems. As a design language,* Hermes/dl *is made up of a collection of primitives, a set of organizing principles, and collections of qualifying situations. A graphical notation shields users from traditional algebraic notation, making* Hermes/dl *more accessible and appealing to potential users in creative and scientific fields. A rigorous graph theoretic formulation will enable mathematical manipulations, and will underlie productivity and collaboration tools that assist users in the design and creation process.* Hermes/dl *implements a new model of computation that emerged from my previous research in software architecture.*

*I will illustrate the use of the concepts developed in* Hermes/dl *with the design and realization of* Mimi, *a multi-modal interactive musical improvisation system that explores the potential and powerful impact of visual feedback in performer-machine interaction.* Mimi *is a performer-centric tool designed for use in performance and teaching. Its key and novel component is its visual interface, designed to provide the performer with instantaneous and continuous information on the state of the system. For human improvisation, in which context and planning are paramount, the relevant state of the system extends to the near future and recent past. The* Mimi *system, designed and implemented using the SAI framework (*Hermes/dl*'s precursor), successfully integrates symbolic computations and real-time synchronization in a multi-modal interactive setting.* Mimi*'s visual interface allows for a peculiar blend of raw reflex typically associated with improvisation, and preparation and timing more closely affiliated with score-based reading.* Mimi *is not only an effective improvisation partner; it has also proven itself to be an invaluable platform through which to interrogate the mental models necessary for successful improvisation.*

*Finally, I will come full circle and argue for the general relevance of the computation model developed.*

*Acknowledgements:* Mimi *is a collaborative project with Elaine Chew (USC Viterbi School of Engineering) and Dennis Thurmond (USC Thornton School of Music).*

**Alexandre R. J. François** est professeur-assistant de recherche en informatique au Computer Science à la Viterbi School of Engineering de l' université de Californie du Sud. Il a été chercheur associé de 2001 à 2004 auprès de deux instances de l'USC, à l'Integrated Media Systems Center et à l'Institute for Robotics and Intelligent Systems. Il est diplômé de l'Institut National Agronomique de Paris-Grignon (France) en 1993, puis obtient, en 1994, le Diplôme d'Etudes Approfondies (DEA) à l'université Paris IX - Dauphine (France) et, respectivement, les tîtres de MS (Master of Science) et PhD (Doctor) en informatique, à l'université de Californie du Sud, en 1997 et 2000.
Ses domaines de recherche couvrent la question englobant la représentation et la manipulation du data et de la connaissance. Son activité de recherche est recentrée sur le thème de conception des systèmes (de logiciel) complexes qu'il a eu l'occasion d'explorer précédemment, dans le contexte des systèmes temps réel complexes, interdisciplinaires et interactifs.

***Alexandre R. J. François*** *is a Research Assistant Professor of Computer Science in the Viterbi School of Engineering at the University of Southern California. From 2001 to 2004 he was a Research Associate with the Integrated Media systems Center and with the Institute for Robotics and Intelligent Systems, both at USC. He received the Diplôme d'Ingénieur from the Institut National Agronomique Paris-Grignon (France) in 1993, the Diplôme d'Etudes Approfondies (M.S.) from the University Paris IX - Dauphine (France) in 1994, and the M.S. and Ph.D. degrees in Computer Science from USC in 1997 and 2000 respectively. His research interests cover all aspects of data and knowledge representation and manipulation. His research activity has centered on the theme of complex (software) systems design, which he has explored in the past in the context of robust, real-time vision systems, and complex, cross-disciplinary interactive systems.*

## ■ 18H15-19H

## DISCUSSION ET SYNTHÈSE
-

**Gérard Assayag** - modérateur

## ■ *6:15PM-7PM*

## *DISCUSSION AND CONCLUSIONS*
-

***Gérard Assayag*** *- Moderator*

# IRCAM

## INSTITUT DE RECHERCHE ET COORDINATION ACOUSTIQUE/MUSIQUE

Fondé en 1970 par Pierre Boulez, l'Ircam est un institut associé au Centre Pompidou et dirigé par Frank Madlener depuis janvier 2006. Il est aujourd'hui l'un des plus grands centres de recherche publique dans le monde dédié à la recherche scientifique et à la création musicale. Plus de 150 collaborateurs contribuent à l'activité de l'institut (compositeurs, chercheurs, ingénieurs, interprètes, techniciens…).

*Founded in 1970 by Pierre Boulez, IRCAM is an institution linked with the Centre Pompidou and has been directed by Frank Madlener since January 2006. Today, it is one of the world's largest public research centers dedicated both to scientific research and musical expression. More than 150 staff members contribute to the institute's activities (composers, researchers, engineers, performers, technicians…).*

L'Ircam est un des foyers principaux de la création musicale de la deuxième moitié du XXe siècle ainsi qu'un lieu de production et de résidence pour des compositeurs internationaux. L'institut propose une saison riche de rencontres singulières par une politique de commandes. De nombreux programmes d'artistes en résidence sont engagés, aboutissant également à la création de projets pluridisciplinaires (musique, danse, vidéo, théâtre et cinéma). Enfin, un grand festival annuel AGORA, permet la présentation de ces créations au public.

*IRCAM is home to musical expressions in all their forms from the second half of the 20th century as well as being a production location and a unique residence for international composers. The institute's season is full of unique encounters with composers and artists from the contemporary stage and it supports young composers with a commission policy. Numerous artist-in-residence programs result in the creation of multi-disciplinary projects (music, dance, video, theater and film). Finally, an important annual festival, AGORA, makes contemporary music creation available to the public.*

L'Ircam est un centre de recherche à la pointe des innovations scientifiques et technologiques dans les domaines de la musique et du son. Partenaire de nombreuses universités et entreprises internationales, ses recherches couvrent un spectre très large : acoustique, musicologie, ergonomie, cognition musicale. Ces travaux trouvent des applications dans d'autres domaines artistiques comme l'audiovisuel, les arts plastiques ou le spectacle vivant, ainsi que des débouchés industriels (acoustique des salles, instruments d'écoute, design sonore, ingénierie logicielle…). Ils sont restitués publiquement à la communauté scientifique lors des rencontres annuelles RÉSONANCES.

*In the realm of music and sound, IRCAM is on the cutting edge of scientific and technocal innovations. Research, carried out in partnership with several universities and international companies, covers a broad spectrum of scientific disciplines; acoustics, musicology, ergonomics, and musical cognition. Ircam's scientific findings are often applied to other artistic domains (audiovisual, fine arts, or live performances) as well as to diverse fields in the industrial world such as room acoustics, live performance technologies, listening devices, sound design, and software engineering. These findings are presented to the scientific community during Résonances, an annual convention on musictechnologies.*

L'Ircam est un lieu de formation à l'informatique musicale. Son Cursus et ses stages réalisés en collaboration avec des chercheurs et compositeurs internationaux font référence en matière de formation professionnelle. Ses activités pédagogiques concernent également le grand public grâce au développement de logiciels pédagogiques et interactifs nés d'une coopération étroite avec l'Education nationale et les conservatoires. L'Ircam s'est enfin engagé dans des formations universitaires avec l'université Paris 6 pour un Master.

*IRCAM is also a center for computer-music education. The institute is a reference point for professional training thanks to its Cursus program and workshops carried out in collaboration with researchers and composers from different countries. Ircam has expanded its educational activities to include the general public by developing interactive teaching software programs in collaboration with the French Ministry of Education and music conservatories.*

**www.ircam.fr**

**www.ircam.fr**

Tickets+Information
+49 (0)30 254 89 100
www.berlinerfestspiele.de

Berliner Festspiele in cooperation
with Stiftung Berliner Philharmoniker

musikfest berlin
August 31st – September 16th
07

Philharmonie
Radialsystem V
Staatsoper
Unter den Linden

Pellegrini Quartett
Concertgebouworkest Amsterdam
Symphonieorchester des Bayerischen Rundfunks
Boston Symphony Orchestra
Konzerthausorchester Berlin
San Francisco Symphony
Ensemble Modern | Staatsoper
Berliner Philharmoniker
musikFabrik | IRCAM Paris
Deutsches Symphonie-Orchester Berlin
Rundfunk-Sinfonieorchester Berlin
Philharmonia Orchestra London
Sächsische Staatskapelle Dresden
Staatskapelle Berlin

Berliner Festspiele

L'Ircam, association loi 1901, est subventionné par le ministère de la Culture et de la Communication (Direction des affaires générales, Mission de la recherche et de la technologie et Direction de la musique, de la danse, du théâtre et des spectacles).

LE FESTIVAL AGORA 2007 EST ORGANISÉ EN PARTENARIAT AVEC :

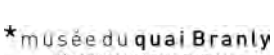
Les Spectacles vivants-Centre Pompidou.
La Cité de la musique
Le musée du Louvre
Le musée du quai Branly
Radio France
Le Théâtre Nanterre-Amandiers
La Ville de Paris
Mairie du IVe arrondissement

AVEC LE SOUTIEN DE :

Ministère de l'Éducation nationale
Caisse des Dépôts
Fondation Calouste Gulbenkian
Pro Helvetia-Fondation suisse pour la culture
Sacem (Société des auteurs, compositeurs et éditeurs de musique)
SACD (Société des auteurs, compositeurs, dramatiques – Action culturelle)
L'Ircam est membre du Réseau Varèse, Réseau européen pour la création et la diffusion musicales, subventionné par le Programme Culture 2000 de l'Union Européenne

L'Ircam remercie ses partenaires médias

Télérama, France musique



# NOTES

–