# THÈSE

présentée à

**L'ÉCOLE POLYTECHNIQUE**

pour obtenir le titre de

**DOCTEUR DE L'ÉCOLE POLYTECHNIQUE**

spécialité :

**INFORMATIQUE**

par

**Cédric FOURNET**

Sujet de la thèse :

## LE JOIN-CALCUL : UN CALCUL POUR LA PROGRAMMATION RÉPARTIE ET MOBILE

*The Join-Calculus:*
*a Calculus for Distributed Mobile Programming*

Soutenue le 23 Novembre 1998 devant le jury composé de :

| | | | |
|---|---|---|---|
| MM. | **Robin** | **Milner** | *Président* |
| | **Roberto** | **Amadio** | *Rapporteurs* |
| | **Gérard** | **Boudol** | |
| | **Jean-Jacques** | **Lévy** | *Directeur de thèse* |
| | **Gérard** | **Berry** | *Examinateurs* |
| | **Luca** | **Cardelli** | |
| | **Georges** | **Gonthier** | |

## Remerciements

– M. Robin Milner a bien voulu présider le jury ; je l'en remercie chaleureusement.
– MM. Roberto Amadio et Gérard Boudol ont lu cette thèse en détail et ont accepté d'en être les rapporteurs. Qu'ils en soient remerciés, et qu'ils me pardonnent ses longueurs.
– M. Jean-Jacques Lévy a été un directeur de thèse amical, disponible, et de bon conseil. Il m'a persuadé de l'intérêt de la recherche en informatique et m'a suggéré l'étude de la programmation répartie. Au cours de la thèse, il m'a apporté son soutien tout en me laissant une grande liberté. Je lui en suis particulièrement reconnaissant.
– M. Georges Gonthier a également guidé cette thèse. Sa collaboration fut essentielle à la plupart des résultats décrits ici. Je l'en remercie tout particulièrement.
– MM. Luca Cardelli et Gérard Berry ont accepté de participer au jury. Je les remercie de l'attention qu'ils accordent à mon travail.

Je tiens à exprimer ma gratitude envers Martin Abadi, Michele Boreale, Georges Gonthier, Cosimo Laneve, Jean-Jacques Levy, Luc Maranget, et Didier Rémy, avec qui j'ai eu la chance de collaborer dans l'étude du join-calcul. Je remercie encore Luc Maranget, qui a réalisé avec moi une implémentation répartie du join-calcul. Notre prototype n'aurait sans doute pas abouti sans son bon sens ni son expérience de la compilation. Peter Sewell et Carolina Lavatelli ont relu le manuscript. Je les remercie de leur patience.

J'ai bénéficié de vives discussions sur les langages de programmation, les calculs de processus, et leur application à la programmation parallèle ou répartie. Merci à Ilaria Castellani, Damien Doligez, Andrew Gordon, Florent Guillaume, Matthew Hennessy, Kohei Honda, Ole Jensen, Fabrice Le Fessant, Xavier Leroy, Ugo Montanari, Uwe Nestmann, Benjamin Pierce, Jon Riecke, Davide Sangiorgi, Peter Sewell, David Turner, et à ceux que j'oublie ici.

Cette thèse s'est déroulée dans l'excellent environnement scientifique et la bonne ambiance du projet PARA et des projets voisins, à l'Institut National de Recherche en Informatique et en Automatique (INRIA). J'ai été invité par MM. Robin Milner et Benjamin Pierce à l'université de Cambridge pendant l'été 1995. J'ai aussi bénéficié des programmes de recherche européens ESPRIT Basic Research Action 6454 - CONFER, puis ESPRIT Working Group 21836 - CONFER-2.

Ce document, ainsi que l'implémentation répartie du join-calcul, ont été produits à l'aide des logiciels Emacs, TeX , LaTeX , et Objective Caml.

# Contents

# List of Figures

# Main Notations

In this dissertation, we use the following standard conventions:

**Tuples:** $\widetilde{v}$ is the tuple $v_1, v_2, \ldots, v_n$ for some integer $n \geq 0$; when the notation occurs several times in the same formula, we assume a different choice of $n$ for each argument of the notation.

**Multisets:** We use the same notations for both sets and multisets—the context should prevent any ambiguity. We write $\{\widetilde{x}\}$ for the set or the multiset that contains the $\widetilde{x}$'s, and $\{x \mid p(x)\}$ to denote the set or the multiset that contains all the $x$'s such that the predicate $p(x)$ holds. We use the operators $\cap$, $\cup$ and, $\uplus$ to note intersection, union, and disjoint union of sets and multisets, respectively.

**Relations and predicates:** A binary relation $\mathcal{R}$ between the sets $S_1$ and $S_2$ is a subset of $S_1 \times S_2$. Most of our relations will range over the same sets. We usually adopt an infix notation for all relations. We let the variables $\mathcal{R}, \mathcal{R}', \varphi$ and symbols of equality and equivalences range over relations. We write $\mathit{Id}$ for the identity relation.

Let $\mathcal{R}$ and $\mathcal{R}'$ be two relations. When defined, we write $\mathcal{R}\mathcal{R}'$ for the composition of relations $\{(x, y) \mid \exists z, x \; \mathcal{R} \; z \; \mathcal{R}' \; y\}$, $\mathcal{R}^{-1}$ for the converse relation $\{(y, x) \mid x \; \mathcal{R} \; y\}$, $\mathcal{R}^n$ for the repeated relation inductively defined by $\mathcal{R}^0 = \mathit{Id}$ and $\mathcal{R}^{n+1} = \mathcal{R}\mathcal{R}^n$, $\mathcal{R}^=$ for the reflexive closure $\mathit{Id} \cup \mathcal{R}$, $\mathcal{R}^+$ for the transitive closure $\bigcup_{n \geq 1} \mathcal{R}^n$, and $\mathcal{R}^*$ for the reflexive-transitive closure $\bigcup_{n \geq 0} \mathcal{R}^n$.

A preorder is a transitive reflexive relation; an equivalence is a symmetric transitive reflexive relation. A relation refines another relation when it is included in it.

Every relation $\mathcal{R}$ defines a predicate, also written $\mathcal{R}$, defined as $x \; \mathcal{R}$ iff $\exists y \mid x \; \mathcal{R} \; y$. For every predicate $T$, the predicate $\not{T}$ is its negation.

**Strings:** A string is a finite sequence of elements from a base set. We let the variables $\varphi, \sigma$ range over strings. $\varphi\sigma$ is the concatenation of the strings $\varphi$ and $\sigma$, and $\epsilon$ is the empty string.

**Variables and Substitutions:** We use variables to denote names and provide scoping rules that determine when a variable is bound. Bound variables can be substituted for any variable that does not appear in its scope; we call such internal substitutions $\alpha$-conversion. A variable is *fresh* with regards to a collection of terms when it does not appear in any of these terms.

We use a postfix notation for substitutions. We write $\{^{y_1}/_{x_1}, \ldots, ^{y_n}/_{x_n}\}$ or simply $\{^{\widetilde{y}}/_{\widetilde{x}}\}$ for the substitution that simultaneously replaces every occurence of a variable $x_i$ by the variable $y_i$, and write $\sigma$ for an arbitrary substitution. We assume implicit $\alpha$-conversion on bound variables before substitution to avoid name clashes.