PPS Activity Report 2007-2012

Laboratoire Preuves, Programmes, Systèmes, UMR 7126
Université Paris Diderot and CNRS

March 1, 2013

Contents

1 General presentation, objectives and results .............................................. 2
   1.1 Themes ......................................................................................... 3
   1.2 Milestones ................................................................................. 7
   1.3 Structure and scientific organization of the PPS lab ............................ 9
   1.4 Scientific animation .................................................................... 11
   1.5 Funding ....................................................................................... 12
   1.6 Evolution of the composition and recruitment policy ............................ 13
   1.7 Summary of results ...................................................................... 16
   1.8 Conclusion .................................................................................. 17

2 Theme: Proofs and Types ............................................................................. 17
   2.1 Lambda-calculus and extensions .................................................... 17
   2.2 Subtypes and intersection type for XML data processing .................... 18
   2.3 Rewriting and security .................................................................... 19
   2.4 Development of the Coq system .................................................... 19
   2.5 Formalization in Coq and in other proof assistants ............................. 21
   2.6 Certification of cost annotating compilers ....................................... 22
   2.7 Dependent types ........................................................................... 22
   2.8 Classical logic, operational duality and effects .................................. 23
   2.9 Delimited control .......................................................................... 24
   2.10 Tensorial proof-nets and combinatorics of proofs ............................. 24
   2.11 Resource control in higher order languages .................................... 25
   2.12 Resources in differential linear logic ............................................. 26
   2.13 Deep inference ............................................................................. 27

3 Theme: Mathematical Foundations of Programming .................................. 27
   3.1 Game semantics and interactive models of computation ....................... 27
   3.2 Games, Ludics and Concurrency .................................................... 28
   3.3 Vectorial models of differential linear logic ...................................... 29
   3.4 The structure of exponentials ....................................................... 30
   3.5 Models of the pure λ-calculus ....................................................... 30
   3.6 Domain theoretic accounts of randomness and non determinism .......... 31
The laboratory Preuves, Programmes et Systèmes (PPS) was created as a joint CNRS and Université Paris Diderot research group in 1999. Its ambition was to gather people from two different cultures: computer scientists coming mainly from the Laboratoire d’Informatique de l’École Normale Supérieure (ENS Ulm) and logicians coming mainly from the Équipe de Logique (Université Paris 7, now Université Paris Diderot). This scientific enterprise, initiated by Pierre-Louis Curien, was driven by a common interest and passion for a better understanding of the mathematical foundations of program and proof construction. Indeed, one of the main unifying paradigms underlying this project is the Curry-Howard correspondence which reveals an isomorphism between computer programs and mathematical proofs.
Since those early days, the PPS lab has grown, exponentially at the beginning and more quietly later, naturally enlarging its scientific interests and applicative ambitions. Despite this extended span of interest, the PPS lab has preserved a unified point of view and a common interest in understanding the structure of programming in the large. For this purpose, we use concepts and tools coming from theoretical computer science and mathematical logic of course, but also from other mathematical theories such as category theory, algebraic topology, probability theory and functional analysis. Our goal is not only to use these theories in our research; we also want to develop a new object of study for mathematicians. In that respect, we believe that Computer Science should play, w.r.t. Mathematics, the same role as Physics has since the 19th century. This is why we insist on maintaining strong institutional relations with mathematicians, and Université Paris Diderot is a perfect institution for hosting such interdisciplinary efforts. At the same time, the PPS lab promotes a unified view of Computer Science, where theory is nurtured by practice and conversely. This is why we are keen to encourage within the lab an experimental approach oriented towards the modelization of complex phenomena like those occurring in implementations of concurrency, computer networking, large software distributions and biological systems.

This integrated approach to Computer Science also justifies the choice of an informal and open organisation of the PPS lab which facilitates internal communication and thematic mobility (see Section 1.3). After internal discussions, we decided to organise this activity report and the associated research project along three federating themes:

- proofs and types
- mathematical foundations of programming
- system modeling, analysis and design

which reflect the current scientific structure of the PPS lab.

1.1 Themes

Proofs and types. The Curry-Howard (CH) correspondence is one of the main methodological guidelines of the PPS lab\(^1\). It works at the three levels of programming languages, since it relates (1) types to logical formulae (2) programs to proofs and (3) program execution to cut-elimination. Initially limited to functional programming languages and intuitionistic logic, this correspondence has been considerably extended since its discovery in the 1970s. Most notably, it was extended to Classical Logic by T. Griffin in the early 90’s. This extension of the CH correspondence was a huge surprise because general cut-elimination in the Gentzen classical sequent calculus is non-deterministic and induces a trivial equivalence relation on proofs: all proofs of the same formula are identified. Strikingly this extension of CH is based on a previously existing construct of the scheme programming language: the call with current continuation effect. A careful analysis led to the discovery that the computational meaning of classical logic relies in fact on a fundamental duality between programs and environments. This discovery is important, because it provides a properly logical account of evaluation strategies in the \(\lambda\)-calculus and in its classical extensions – call by name versus call by value in particular – as well as a logical understanding of advanced programming effects like delimited control. This unexpected success also motivates the study, from

\(^1\)An important proportion of the research presented in the Proofs and Types and Mathematical foundations of programming themes is supported by the ANR blanc project RECRE \url{http://recre.ens-lyon.fr/}. 
a logical point of view, of other extensions of the lambda-calculus e.g. with patterns or with explicit substitutions — motivated by programming practice — which are now very well understood from the operational point of view, and admit clean syntactical presentations.

This duality between programs and environments was observed for the first time in Linear Logic (LL) where it is expressed through the notion of polarity on formulae, and polarity-based restrictions on the use of the structural rules. This dual point of view enables a reduction of the proof-search space by focalizing proofs, that is, by gathering the rules of a derivation tree into clusters of the same polarity. This combinatorial idea is the starting point of ludics, an interactive presentation of logic deeply related to game semantics. In that way, LL provides a nice unifying framework, which combines an elegant categorical semantics with an asynchronous representation of proofs (proof-nets and interaction nets) supported by a purely algebraic description of resources investigated by differential linear logic.

Our foundational advances would not have been possible without a strong involvement of the PPS lab in the current practice of typed programming languages. Types are particularly important in programming because they guarantee a correct execution and a sound control of resources. The development of more expressive type systems including polymorphism, subtyping and intersection types is thus a requisite in order to manipulate more safely the complex tree-like structures of XML, as done in the CDuce programming language. Great care has been also devoted to the development of region calculi for multi-threaded programs with effects, and to complexity-bounded programming languages, both using ideas coming from LL.

The PPS lab has also benefited from the creation of the INRIA team pi.r2, this integrating the development and the maintenance of the Coq proof assistant as a major activity of the lab. Coq provides a sophisticated environment (including tactics) for writing mathematical proofs and certifying software properties. As a language, Coq is a higher order extension of Martin-Löf Type Theory extended with inductive and co-inductive types. It comes equipped with a rich collection of libraries implementing various mathematical theories. It has many applications in mathematical proof formalization of course (e.g. the Four Colors Theorem formalized in Coq by Gonthier and Werner) but also in program certification (Leroy’s C-minor compiler certified in Coq). The PPS lab is also involved in the construction of certified compilers, providing certified bounds on the complexity of the generated low-level code.

Recently, proof formalisms generalizing the sequent calculus by allowing rules to be applied anywhere in formulae — and not only at the outermost level — have been introduced under the generic name of deep inference. These systems have a strong categorical flavour and allow the representation of logical rules which are out of the scope of the more standard sequent calculus, natural deduction and proof net formalisms. There, the main issue is to understand cut-elimination and to relate these formalisms with more usual proof representations.

Mathematical foundations of programming. The starting point of denotational semantics is the discovery that λ-terms can be seen as proofs (through the CH correspondence) and at the same time as (continuous, stable...) functions between the relevant spaces. The observation that they may be also represented as linear or analytic maps between topological vector spaces opened the study of their differential calculus, leading to differential linear logic (DiLL) and a generalized Taylor expansion formula. These topological models are based on a new understanding of the relationship between topology and computation, distinct from the traditional domain-theoretic view discovered by Scott. In particular, they rely on the fundamental duality between bounded and open subsets of functional analysis. These ideas provide firmer foundations for resource sensitive extensions
of the λ-calculus introduced in the 1990’s after Milner’s π-calculus. They enable to improve the syntax of these calculi, to clarify their operational semantics and to develop an analytic approach to termination. In this way, they offer a major bridge between syntax and semantics, suggesting moreover new developments towards probabilistic models.

This cross-fertilization between semantics and syntax is a distinctive feature of the scientific methodology of the PPS lab. It is also illustrated by the introduction of tensorial logic, a refinement of LL where linear negation is not involutive, motivated by a careful analysis of game semantics, where proofs and programs are interpreted as interactive strategies playing on dialogue games. Tensorial logic provides a type-theoretic foundation to game semantics, and enables to see strategies as tensorial proof-nets formulated in the algebraic language of string diagrams.

These investigations at the interface of syntax and semantics are also justified by our desire to understand the algebraic structure hidden in Jean-Louis Krivine’s classical realizability. This classical version of Kleene realizability offers a fundamental bridge between programming languages and Cohen forcing. In particular, it enables to construct a model of Zermelo-Fraenkel set theory from any functional language extended with imperative features. In this framework, every formula becomes the specification of untyped imperative programs formulated in a completely independent fashion. In particular, and quite surprisingly, it appears that well-known axioms of set theory (like the axiom of dependent choice or the continuum hypothesis) specify existing imperative effects like memory access or the quote command of scheme. A central purpose and achievement of classical realizability is to characterize the various realizers of a given formula (typically an axiom) by exhibiting their common operational behaviour.

Rewriting theory is another key component of our toolbox in the mathematical investigation of computations. Whereas denotational and game semantics describe the potential interactions of a program with its environment, rewriting treats programs as intrinsically dynamic objects. From this point of view, the main issues are termination (or, more generally, responsibility) and confluence. Logical tools such as reducibility (a variant of realizability) are essential for dealing with termination. One main conceptual step concerning confluence has been the discovery of a deep relationship with the so-called coherence theorems established in higher dimensional algebra. This new geometric understanding of confluence as the first step (dimension 1) of a higher-dimensional ladder also including standardization (dimension 2) leads to the general question of adapting well-known techniques of rewriting theory (like the Knuth-Bendix completion or termination orders) to handle these higher-dimensional rewriting systems. This question also opens promising connections with mainstream mathematics, this including homotopy (around Quillen categories), homology (around the Squier Theorem) and computational algebra (around Gröbner bases).

We believe that these geometric tools related to homotopy theory will play an increasing and unifying role in the future of programming languages and more generally, computational systems. They already appear as a fundamental tool in the study of concurrent and distributed processes. The idea is to consider critical sections (typically coming from locks on shared resources) as geometric obstructions, leading to non-trivial homotopy invariants of the associated computational space. Quite surprisingly, homotopy theory also appears in Martin-Löf Type Theory, as noticed in the 1990s by Hoffman and Streicher. This pioneering observation followed by Voedvodsky’s impulse has been the starting point of a very active line of research on the homotopic structure of equality types in the proof assistant Coq, and of a debate on the status of the univalence axiom when confronted to proof-irrelevance. Finally, homotopy theory gave rise to operads like A∞-spaces which describe algebraic structures up to topological deformation. It appears that the operadic framework
is tightly connected to the notion of monad used in programming languages like Haskell for dealing with effects and non-functional system calls. In particular, they provide purely combinatorial description of well-known effects like memory calls or exceptions. These algebraic and topological investigations developed in the PPS lab reinforce our interactions with the mathematicians of Université Paris Diderot (IMJ).

**Modeling, analyzing and conceiving systems.** Although programming languages were originally designed in order to implement algorithms, they also provide a powerful tool to model computationally complex situations. This is a key feature of Milner’s π-calculus, which is designed as a foundational concurrent programming language, and at the same time as a process algebra reflecting the temporal and resource conflicts between distributed agents. The study of these process calculi is a fundamental activity of the PPS lab, and comes with various purposes, like the design of synchronous versions of the π-calculus, or the type-theoretic accounts of interactions based on session types.

However, these process calculi should be refined in order to describe properly and efficiently biochemical systems like cellular signaling pathways. The κ-calculus was introduced in that purpose: its local and reversible representation of interactions enables to model these chemical systems and to simulate them, typically using formal methods like abstract interpretation in order to improve modularity and efficiency. The concrete case studies have led to interesting and new theoretical questions related to Winskel’s description of causality cascades using event structures. In particular, causality is related to reversibility, a concept which plays a central role in biochemistry, and should become a basic ingredient of concurrent and distributed programming languages, integrated as a semantically valid undo primitive.

Many modeling situations like biochemical systems are of an intrinsically stochastic nature. The PPS lab is thus involved in a long-term mathematical investigation of new probabilistic and concurrent models of computation, offering an appealing unification of event structures and Markov chains. An important outcome of the approach is the emergence of rigorous limit theorems in concurrency theory.

The development of formal tools in the PPS lab is also propelled by the project of analyzing complex situations encountered in the context of large software distributions. The main difficulty is that these distributions consist of a huge number of small components developed by a crowd of independent programmers — this is mostly true of the open source systems in which the PPS lab is involved — with complex dependency and incompatibility relations between them. When a whole system, or even a simple package, is installed on an actual computer, it is of course essential to take into account these dependencies and incompatibilities. This crucial analysis is computationally complex and requires clever data structures and efficient algorithms.

As a research lab devoted to programming languages, the PPS lab integrates the design and use of domain-specific programming languages (typically extending C or Ocaml) as one of its main activities. Concurrent programming is a particularly important field since most operating systems are concurrent and difficult to program and to debug. The programming concept of thread offers a very pleasant and comfortable framework but notoriously difficult to compile. The CPC translator was designed to turn thread-based programs into programs which use event-loops — much more difficult to write directly without mistakes, but much easier to compile. This technological achievement is based on a continuation passing style translation, derived from Gödel’s translation of classical logic in intuitionistic logic. The PPS lab is also involved in the design and implementa-
tion of the Ocsigen web server, together with the Eliom web programming environment\textsuperscript{2}. Together, they allow to combine the best of traditional and persistent applications, allowing to write them in a classical declarative way and offering the power of the OCaml type system to enforce many properties at compile time. A striking benefit of the language is to ensure that Javascript APIs are correctly used, that database accesses are safe, that links go to existing services with the right parameters. These achievements rely on an efficient implementation of light threads also performed in the PPS lab, and integrated in the OCaml Lwt library.

1.2 Milestones

The period under examination has been marked by several important events for the PPS lab. Some of them have deeply modified the structure of the laboratory.

Creation of the pi.r2 team. pi.r2 is an INRIA - Université Paris Diderot - CNRS EPC (équipe-projet commune) created in 2008. It originates from a long term scientific collaboration between Pierre-Louis Curien (CNRS, creator, former director and member of PPS) and Hugo Herbelin (INRIA). Their common interest in classical extensions of the $\lambda$-calculus and in sequent-based calculi is at the origin of one of the scientific themes of pi.r2. Hugo Herbelin is also one of the main architects of the Coq proof assistant, and Pierre Letouzey, a member of the PPS lab since 2005, was already very active in the development of Coq and so it was natural to include this activity in the scientific themes of the new pi.r2 team. pi.r2 is now the leading group for the development and maintenance of Coq, reinforced by the recruitment of Matthieu Sozeau (INRIA) in 2010. Of course, all the scientific activities of pi.r2 are deeply related with those of the rest of PPS.

The pi.r2 team is hosted by INRIA, in its building located avenue d’Italie. Scientific interactions between these two geographically separated parts of the PPS lab have been reinforced by the creation of a common working group on type theory and realizability. When the PPS lab will move to the Sophie Germain building at the beginning of 2013, the pi.r2 group will also move there, thus joining the rest of the lab.

Creation of IRILL. Open source is an increasingly significant model of software development which has many specificities and advantages. It opens markets which were closed and captured by a few companies. It allows software users and producers to have complete control of, and to trust, the tools they use, because they have free access to the source code. Moreover, open source has a social and political significance and impact since this software becomes a common heritage of all of society instead of private goods. Open source software is also a source of new fascinating scientific questions and educational challenges. In order to address these issues, and after an already rich experience in research on open source acquired in the EDOS and MANCOOSI European projects, Roberto Di Cosmo (PPS) decided to promote the creation of a new research structure devoted to this topic. His efforts have been crowned with success in 2010, with the official inauguration of the Initiative de Recherche et d’Innovation sur le Logiciel Libre (IRILL), a joint INRIA, Université Paris Diderot and Université Pierre et Marie Curie structure. IRILL is not an autonomous research group; its purpose is to host visitors, organize meetings and seminars and to stimulate the cooperation between private companies and academic research groups; accordingly,

\textsuperscript{2}Ocsigen is now used by several companies, like BeSport or MetaWeb (both in San Francisco), Li-Cor (Lincoln, Nebraska), Hypios (Paris), etc. Several startups are currently being created using the framework. Of course, the web site of the PPS lab is hosted by an Ocsigen server.
IRILL has permanent contacts with the SYSTEMATIC Paris Région competitiveness pole, through its groupe thématique sur le logiciel libre (GTLL).

**Creation of the INFINIS international laboratory.** The long-term scientific cooperation\(^3\) between France and Argentina in Computer Science led to the creation of the International Laboratory (LIA) INFINIS, at the end of 2011. This creation is the result of the constant efforts of Delia Kesner (PPS) for pursuing this project over the last five years. She shares with Sergio Yovine (CONICET) the directorship of this LIA in which Université Paris Diderot and CNRS in particular are involved and which will be an efficient tool for enforcing our already rich scientific interactions with South America.

**Creation of the Labex Sciences Mathématiques de Paris.** Since its creation, the PPS lab has been an active member of the FSMP (*Fondations Sciences Mathématiques de Paris*), an RTRA (*Réseau Thématique de Recherche Avancée*) which gathers all mathematics labs of Paris *intra muros* as well as the PPS lab and the LIAFA. The PPS lab was actually one of the founding members of this RTRA in 2005, and Pierre-Louis Curien has been one of its co-directors until the creation of the Labex. In 2008, the FSMP decided to apply to the competition launched by the Ministry of Research for the *Laboratoires d’Excellences* (Labex), whose purpose was to gather high level research teams around well defined scientific goals and projects. During the preparation of this application, we insisted on the importance of Computer Science among Mathematical Sciences and on its specificities. This led to the inclusion of the *Laboratoire d’Informatique de l’École Normale Supérieure* (LIENS) in the Labex *Sciences Mathématiques de Paris* which won the competition in 2010, and to a number of programmes devoted to Computer Science in this Labex.

The FSMP has developed many innovative programmes which have reinforced the international visibility of the Mathematics and Fundamental Computer Science departements of Paris. Among them, we can outline a post-doc program from which the PPS lab has regularly benefited, and the Paris Graduate School program whose purpose is to attract top students to our Masters and PhDs.

**Creation of two startup companies.** Balat is currently creating a startup OCSIGEN Lab for developing Web applications with Ocsigen and helping Ocsigen based projects to emerge. This project is candidate for the *concours national de création d’entreprise - Oséo (catégorie “émergence”)*. It is supported by INRIA and IT Translation and is candidate for being supported by the SATT\(^4\) Île de France Innov. Balat is also part with Vouillon (PPS) and Pierre-Laurent Jouannet (Teikhos) of a project for creating a company for file synchronization, based on the Unison file synchronizer and the Ocsigen framework. This project is candidate for the *concours national de création d’entreprise - Oséo* (catégorie “création développement”).

**Preparing our move to the new building Sophie Germain.** Since its creation, a major concern for the PPS lab has been its constant lack of physical space. Located in a building *Rue du Chevaleret* which is not managed by Université Paris Diderot, the laboratory has had to reconfigure various times since its arrival in 2000. In particular a major internal reorganization of the offices of the PPS lab has been performed in 2010 after the departure of the mathematicians affiliated to Université Pierre et Marie Curie. New buildings have been constructed by Université Paris Diderot located on its campus in the *Paris Rive Gauche* area, close to *Bibliothèque François Mitterrand*. At

---

\(^3\)Concerning PPS, this cooperation took place in particular within the STIC-AMSUD program (*Types for Robust Program Development* project) and within the CNRS-CONICET projects *Langages avec Motifs: Syntaxe et Sémantique* and *Réécriture axiomatique*.

\(^4\)SATT: Société d’Accélération de Transfert de Technologie.
the beginning of 2013, the Sophie Germain building will host the entire Mathematics and Computer Science departments of the University. Organizing this move has been a delicate task and we arrived at a configuration where the PPS lab will share the 3rd floor of the building with the Computer Science UFR administration, close to the LIAFA (4th floor). With respect to the actual situation, this move will thus provide a welcome improvement to our working environment. However, the main benefit from our point of view is that the PPS lab and the pi.r2 team will be located at the same physical place.

A new computer infrastructure. The planning of this move has been combined with a comprehensive reorganization and rationalization of the computer services of PPS. Since the building Sophie Germain has a single computer room, it was natural to gather the computer services of the various departments located in the Sophie Germain building into a single technical team, with a common computer infrastructure. This has been carried out over the past 5 years by Yves Legrandgérard (PPS) together with his colleagues from the Mathematics Department, leading to a common platform based on modular hardware, benefiting from the newest virtual machines technologies and offering an increasing number of services\(^5\). This common infrastructure is operational and ready for transfer to the new building, and is supervised by the Fédération de Recherche en Mathématiques de Paris Centre (FR 2830) of which PPS is a member.

The project of a joint LIAFA-PPS lab. In July 2010, the Computer Science Institute of CNRS (INS2I) asked the two directors of LIAFA and PPS to consider the idea of unifying their laboratories, ideally on the horizon of 2014. The main objective was to create one large Computer Science Laboratory in Université Paris Diderot which would span almost all the spectrum of Fundamental Computer Science. In order to examine the various benefits of such a merge, but also the difficulties and issues that we would have to overcome in order to achieve this goal, a LIAFA-PPS committee headed by Pierre Fraigniaud was created and met regularly during the year 2011. A detailed report summarizing its conclusions was released at the beginning of 2012 and submitted to the members of the two laboratories. A general meeting of each lab was organized the same week in order to stimulate free discussions about the benefits and dangers of this potential fusion. One main concern during these discussions was the difference between the architectures of the two labs. The idea of building a joint lab was openly supported by the directors of LIAFA and PPS, who explained their position in a joint letter emailed to the members of the two labs. A referendum was organised at the beginning of the following week among the permanent members of each of the laboratories. After the ballot, it appeared that a moderate but clear majority of the PPS members rejected the idea of an immediate merge, whereas a majority of the LIAFA members voted for it. This immediately stopped the administrative process, or at least postponed it, since the idea of a joint LIAFA-PPS lab remains a natural and appealing perspective. As a matter of fact, this long process of discussions revealed that, whatever the way the two labs are articulated administratively, their respective members are convinced, to a vast majority, that they would gain a lot in strengthening their scientific links. This convergence should be supported by a firm scientific policy and encouraged by the organization of common scientific events like seminars, lectures, meetings, or projects. A joint LIAFA-PPS scientific committee integrating members of the IMJ logic team, headed by Paul-André Melliès (PPS) and Olivier Serre (LIAFA), has been created in May 2012 in order to launch and promote such actions.

---

\(^5\)See https://www.math.univ-paris-diderot.fr/sysadmin/.
1.3 Structure and scientific organization of the PPS lab

The *Règlement Intérieur* of the PPS lab is attached to this dossier.

One remarkable and important feature of the PPS lab is that it is not divided into formal teams. This choice of structure comes from history, and is justified today by the general philosophy and the scientific objectives of the lab, described above. It is also made possible by the following facts.

- The PPS lab is still relatively small (about 40 permanent members) and is structured around a series of regular thematic and well-attended working groups where local and external speakers are invited to give technical talks about their research, see Section 1.4. Concretely, the attendees of these working groups constitute flexible research teams within the PPS lab, attracting and naturally integrating external researchers.

- This informal organization is complemented by the fact that several PPS members take part to the joint research team pi.r2 with INRIA, or to the IRILL initiative with INRIA and Université Pierre et Marie Curie (see Section 1.2). These structures provide in particular a favorable framework for supporting several of the major software developments to which PPS contributes, like Coq and Ocsigen.

- In addition, the PPS lab is structured by the many national and international projects (see Section 6) in which it is involved. Each project is organized around clear scientific objectives, with strong scientific animation activities, such as organization of seminars, workshops, conferences. These projects are essential today for funding post-doc researchers and PhD students. They are also particularly important for supporting software developments, see *e.g.* Section 4.7, 4.8 and 4.9.

**Conseil de laboratoire.** The major decisions concerning the PPS lab are discussed by its *Conseil de laboratoire*. Among these decisions, we can mention:

- The hiring policy: request for Professor and Assistant Professor positions opening to the Université Paris Diderot, determination of the scientific profile of these positions, suggestions for the composition of the *Comités de Sélection*.

- PPS post-doc candidates ranking (typically for the FSMP post-doc program).

- Université Paris Diderot, CNRS and FSMP foreign professors invitations: ranking of candidates.

- Preparation for the *Demande de moyens CNRS* in September.

Meetings of the PPS *Conseil de laboratoire* are organized when necessary, roughly every two months. Some of the members of the PPS *Conseil de laboratoire* were elected by the lab in spring 2009 for a period of 4 years. The other members were nominated by the director.

- *Collège enseignants-chercheurs*: Samy Abbes (elected), Delia Kesner (elected), Yann Régis-Gianas (nominated), Daniele Varacca (nominated).

- *Collège chercheurs*: Giuseppe Castagna (elected), Paul-André Melliès (nominated).

- *Collège ITA*: Odile Ainardi.

- *Collège IATOSS*: Yves Legrandgérard.
• **Collège doctorants**: Fabien Renaud.

• **Permanent invited member**: Pierre-Louis Curien.

Since many important issues concerning the PPS lab are also discussed in the *Conseils scientifiques* of the Computer Science Department and of the Mathematics Department (UFR), the members of PPS who participate to them are systematically invited to the meetings of the *Conseil de laboratoire*: Roberto Amadio, Vincent Balat, Roberto Di Cosmo, Gabriel Kernéis, Pierre Letouzey, Jean-Marie Rifflet, Ralf Treinen, Daniele Varacca.

**Specific tasks.** A few committees, internal to PPS, are in charge of specific questions, currently:

• *Commission des moyens*, responsible for the purchase of new computer devices (Ainardi, Legrandgérard, Letouzey, Rozière).

• *Commission des locaux*, responsible of the internal moves of PPS and of its move to the *Sophie Germain* building (Ainardi, Castagna, Ehrhard, Kesner, Legrandgérard).

• *Commission du séminaire*, responsible of the organization of the PPS seminar (Faggian, Jean Krivine, Saurin).

and a few members of the lab have specific responsibilities, currently:

• Legrandgérard is responsible for the computer infrastructure,

• Harmer is CNRS delegate for communication,

• Ainardi is CNRS delegate for training,

• Balat is responsible for the Web site,

• Bucciarelli is responsible for international relations and supervising invitations,

• Vouillon is our HAL correspondent\(^6\),

• Tasson is PhD adviser, meaning that she is in charge of keeping a regular and friendly contact with all the PhD students of the PPS lab, and of helping them in case of difficulties.

• Melliès is responsible for the documentation (internal PPS library) and for the relations with universitary libraries.

### 1.4 Scientific animation

The scientific animation of the PPS lab is implemented by means of a number of specific actions supported by the laboratory and by the associated institutions pi.r2 and IRILL.

• The PPS Seminar\(^7\). It takes place each week, traditionally on Thursday morning in the *Rue du Chevaleret* building and is currently organized by Claudia Faggian, Jean Krivine and Alexis Saurin. The topics chosen are of general interest for the members of the PPS lab and the invited speakers are asked to make as widely accessible presentations as possible.

---

\(^6\)He is in charge of giving the official PPS agreement stamp to publications uploaded on the HAL server by members of the lab.

• PPS regular Working Groups. These are more specialized and less formal seminars, where talks can typically be interrupted by questions of the attendees and longer technical discussions can take place. Members of the PPS lab are frequently invited to present their ongoing work in these working groups. The currently active working groups are:

- Sémantique, organized by Paul-André Melliès and taking place each week in the Rue du Chevaleret building.

- Réalisabilité et théorie des types, organized by Hugo Herbelin and Paul-André Melliès since the creation of the pi.r2 team, and taking place each week in the INRIA Avenue d’Italie building where the pi.r2 team is currently hosted. In addition to its purely scientific interest, one goal of the working group is to provide a scientific forum between the two parts of the PPS lab (Rue du Chevaleret and Avenue d’Italie).

- Programmation, organized by IRILL (Emmanuel Chailloux) and taking place most often in the INRIA Avenue d’Italie building (where IRILL is located), every two weeks.

- Catégories supérieures, polygraphes et homotopie, organized by François Métayer and taking place each week in the Rue du Chevaleret building. This working group is devoted to an emerging topic at the frontier of programming language semantics and homotopy theory, and is attended by PPS members as well as mathematicians members of the IMJ.

The working group Concurrence et programmation aléatoire was active until 2010 and is currently stopped, mainly because its topics are partly covered by the other PPS working groups. We plan to relaunch it in the near future.

• PPS exceptional Working Groups. These working groups are organized from time to time on the initiative of the members of the PPS lab. Typical examples are the working group on classical realizability organized by Paul Rozière in Spring 2011 or the working group on algèbre and lambda-théories organized in Spring 2012 by Thomas Ehrhard, on the occasion of the visit of Antonino Salibra at PPS.

• PPS Annual Meeting. In order to take into account the widening of the scientific span of PPS, it seemed useful to organize a regular internal PPS meeting (two days) with scientific talks given by the members of the lab so as to increase the knowledge of all members of PPS of the activities of the lab. After the success of the first venue in September 2011 organized by Delia Kesner in a hotel of Trouville (more than the half of the lab attended), a second instance has been organized in the same place, again by Kesner, in September 2012.

• Invitation of external researchers (1 month or more). The PPS lab has benefited regularly from CNRS associate researchers positions: one per year, lasting from 3 to 6 months. It has also widely benefited from the Université Paris Diderot invitation program, and from other invitation programs as well (e.g. FSMP and Ville de Paris).

An important event in the national scientific landscape in Computer Science has been the creation by the CNRS of the GDR-IM in 2006. The PPS lab has immediately taken an important animation activity in this GDR, and more specifically in the birth of the LAC (Logique, Algèbre, Calcul) and of the GEOCAL (Géométrie du Calcul) working groups (see 6.2).

\[^{8}\text{See \url{http://www.pps.univ-paris-diderot.fr/gdt}.}\]
1.5 Funding

The general expenses of the PPS lab are funded by yearly subventions received from CNRS and Université Paris Diderot\textsuperscript{9}. The typical amount of these subventions is 35 000€ for CNRS and 70 000€ for Université Paris Diderot, but the global tendency is a regular decrease of these amounts, in spite of the efforts of these institutions to maintain them as high as possible. These resources are extremely useful to fund travels and invitations which cannot be funded by projects or are of global interest for the laboratory. Typical examples of such operations are the PPS Annual Meeting, initiated in September 2011, and the invitation of speakers to the PPS Seminar, who often stay a week or more in the lab. These subventions also enabled us to buy computers for the new PPS members, and to buy servers and other devices as contributions to the new computer infrastructure, see Section 1.2.

These global financial resources are complemented by fundings coming from a significant number of projects of various kinds, listed in Section 6. In this section, the reader will find a detailed description of these projects as well as the funding received by the PPS lab from these projects when this applies (when the funding is not given, this means usually that the project has directly funded travels of PPS members, without transferring money to our accounts).

An essential part of this resource has been used to fund post-doctoral researchers in the PPS lab.

1.6 Evolution of the composition and recruitment policy

The current composition and *organigramme* of the PPS lab is provided as a document attached to this dossier.

**Administrative staff.** Odile Ainardi is the person in charge of the administrative management of the PPS lab since 2001. She has taken up alone all the administrative management of the PPS lab until 2009, in spite of the constant growth of the lab. Pierre-Louis Curien, the former director of PPS, asked the CNRS for another administrative staff position to be attributed to PPS and obtained it in 2008.

In November 2008, Audrey Mansuet was recruited as a CNRS ITN\textsuperscript{10} at PPS\textsuperscript{11}. She started to work in the laboratory under the supervision of Odile Ainardi who helped her to improve her knowledge of all aspects of managing a joint CNRS-university laboratory like PPS, from financial aspects to the management of employees. Audrey Mansuet also attended several training sessions organized by the CNRS on various aspects of these managing activities, and on mastering the associated softwares. In October 2010, she succeeded in the CNRS AI\textsuperscript{12} competition, got recruited as an AI in a biology research laboratory and left PPS in February 2011.

In June 2012, INS2I CNRS Institute attributed to PPS a one year administrative staff position. We recruited Melina Sainte-Rose on this position: she currently provides essential help and assistance to Odile Ainardi.

**Scientific permanent members.** Since January 2007, the following members of PPS have left the laboratory.

---

\textsuperscript{9}As an INRIA team, pi.r2 has its own INRIA funding which is not managed by PPS. Similarly, as an autonomous structure, IRILL has its own funding from INRIA, Université Pierre et Marie Curie and Université Paris Diderot.

\textsuperscript{10}Administrative Technician.

\textsuperscript{11}Odile Ainardi was a member of the selection committee.

\textsuperscript{12}Administrative Engineer.
• Chantal Berline (CNRS Chargée de Recherche, CoNRS 1) retired in October 2008.

• Emmanuel Chailloux (Université Paris Diderot Assistant Professor, CNU 27) left PPS in July 2007 and became Professor at Université Pierre et Marie Curie, and a member of the LIP6 lab.

• Guy Cousineau (Université Paris Diderot Professor, CNU 27) left PPS in June 2012. He is a member of AERES since 2008 (in délégation). Before 2008, he was President of Université Paris Diderot.

• Vincent Danos (CNRS Directeur de Recherche, CoNRS 7) left PPS in September 2008 to become Professor at University of Edinburgh (LFCS). He is now in détachement\textsuperscript{13} until September 2014 and plans to come back to PPS after that date.

• Jean-Baptiste Joinet (Université Panthéon Sorbonne Assistant Professor, CNU 17) left PPS in January 2012, to become a member of the CIRPHLES lab of ENS Ulm.

• Olivier Laurent (CNRS Chargé de Recherche, CoNRS 1) left PPS in August 2008, to become a member of the LIP lab (ENS Lyon), team Plume. This is a mutation, that is, a permanent departure.

• Alexandre Miquel (Université Paris Diderot Assistant Professor, CNU 27) left PPS in September 2009 to become a member of the LIP (team Plume) lab at ENS Lyon where he is détaché.

• Jean-Marie Rifflet (Université Paris Diderot Professor, CNU 27) left PPS in June 2012.

The following members of PPS arrived during the period 2007-2012:

• Samy Abbes arrived at PPS in October 2007 as an Université Paris Diderot Assistant Professor (CNU 25).

• Yves Guiraud arrived at PPS–pi.r2 in June 2012 as an INRIA Chargé de Recherche.

• Hugo Herbelin arrived at PPS–pi.r2 in January 2009 as an INRIA Directeur de Recherche.

• Jean Krivine arrived at PPS in September 2009 as a CNRS Chargé de Recherche (CoNRS 7).

• Yann Régis-Gianas arrived at PPS in September 2008, as an Université Paris Diderot Assistant Professor (CNU 27).

• Paul Ruet arrived at PPS in September 2011 as a CNRS Chargé de Recherche (CoNRS 1).

• Alexis Saurin arrived at PPS–pi.r2 in September 2010 as a CNRS Chargé de Recherche (CoNRS 7).

• Matthieu Sozeau arrived at PPS–pi.r2 in September 2010 as an INRIA Chargé de Recherche.

• Christine Tasson arrived at PPS in September 2009 as an Université Paris Diderot Assistant Professor (CNU 27).

• Ralf Treinen arrived at PPS in September 2007, as an Université Paris Diderot Professor (CNU 27). He was before Assistant Professor at ENS Cachan, member of the LSV lab.

\textsuperscript{13}Meaning that he is still a member of the CNRS, but does not receive a salary from it.
• Stefano Zacchiroli arrived at PPS in September 2011, as an Université Paris Diderot Assistant Professor.

The global balance (+3) is clearly positive for the PPS lab, and a significant part (4) of these arrivals are directly related to the creation of the pi.r2 common team with INRIA. The effect of these arrivals is a reinforcement of the PPS research potential on all of its three research themes described above. This is the result of a long-thought recruitment policy developed jointly by the laboratory and Université Paris Diderot, CNRS and INRIA over the years, and whose purpose is to promote our scientific programme explained above.

**PhD students.** The PPS lab has hosted 48 PhD students during the 2007-2012 period, with a reasonable number of new PhD students each year\(^\text{14}\), mainly coming from the MPRI and LMFI Research Masters. The strong involvement of the PPS members in these masters has been essential for attracting these students. All of these students have been funded by PhD grants of various kinds. Among them, 28 have defended their PhD thesis and 16 are currently working on it. Five PhD students trained in the PPS lab received prestigious national or international awards for their PhD thesis: Carraro (Prize for the best scientific PhD of Ca’Foscari University), Ilik (Gödel Prize), Lengrand (Ackerman Prize), Manzonetto (EADS Foundation) and Tabareau (Prix de la Chancellerie, 2nd Gilles Kahn Prize). Two PhD students also resigned during that period, in order to become software engineers in private companies, and one PhD student decided to change his research topic.

It should be noted that the average number of PhD students hosted by the PPS lab during the period is 18. We believe that this number can be improved considering the scientific dynamic of the PPS lab. One should keep in mind however that after twelve years of creation, the lab remains a group of relatively young researchers with only 12 members having their HDR. One of our main purposes currently is thus to encourage our researchers to pass their HDR and to start supervising PhD students: 2 of them (Melliès and Varacca) will pass their HDR within a few months and 3 others at least (Balat, Chroboczek, Padovani) are clearly on the HDR track. This general policy is supported by the Computer Science UFR with (partial) délégations for the Assistant Professors. These forthcoming HDRs will greatly increase the PhD supervision capabilities of the PPS lab.

An additional issue is that finding PhD grants is more and more difficult outside dedicated research projects. Factually, the PPS lab can expect only 2 or at most 3 PhD grants each year from its Doctoral School (Doctoral School 386), meaning that the other PhD thesis are funded by other means: mainly grants from Écoles Normales Supérieures, from École Polytechnique or from national and european research projects.\(^\text{15}\) As a recent illustration of this issue, the PPS lab obtained in 2012 two grants for four candidates from Doctoral School 386, and thus two very good students wishing to prepare their PhD thesis at PPS and having found their supervisors here had to give up due to a lack of local funding: as a result, PPS should have hosted 8 instead of 6 new PhD students in September 2012. To improve the situation, we will suggest more systematically PPS ANR project promoters to present four-year proposals including PhD grants and we will encourage submission of ERC Research projects.

\(^\text{14}\)Apart for the exceptional year 2007 where 10 students started a PhD at PPS, the typical number of new PhD students is 5 per year.

\(^\text{15}\)The end of the domaine d’intérêt majeur (DIM) Région Île de France Logiciels et Systèmes Complexes (LSC) means also that one of the possible sources of PhD grants has vanished.

\(^\text{16}\)A PhD grant is for 3 years and an additional preleminary year is useful to find a good candidate.
**Post-doctoral researchers.** During the same period, the PPS lab has hosted 26 post-doctoral researchers for periods from 6 months to 3 years (most of them were 12 months positions), 21 among them coming from foreign universities. Most of these positions were funded by projects of various kinds (ANR, European, *Région Île de France* projects). A few of them were directly funded by Université Paris Diderot, CNRS or INRIA. In addition, a series of post-doc positions were funded by the FSMP. These positions are particularly prestigious and lead each year to a severe competition, the candidates being selected by a high-level FSMP committee. The PPS lab obtained 5 such positions during the period 2007-2012 (including a position starting in September 2012).

**Invited researchers.** During the period under consideration, the PPS lab hosted nearly 70 invited researchers. In most cases, the invitations lasted one month and were funded by Université Paris Diderot. In some cases, funding was by the CNRS, by INRIA, by the FSMP or by research projects PPS was/is involved in.

**Training of the members of the PPS lab.** Several members of the lab have taken part to training sessions organized on various topics by the Université Paris Diderot, CNRS or other institutions. Our policy is also to encourage PhD students and young researchers to to take part to Summer Schools and similar events, and to (co)organize such events as we did for *Logic and Interaction 2012* weeks in Marseille.

### 1.7 Summary of results

Sections 2, 3 and 4 provide a reasonably accurate description of the scientific activities of the PPS lab in each of its three themes. In the following sections, we try to provide an exhaustive description of our production and activities from various points of view. We provide here a short summary of this material.

**Publications.** The PPS lab has published 112 international journal papers and 164 peer-reviewed international conference papers during the period 2007-2012, see Section 5. This production is of high quality with a good balance between journal and conference papers.\(^{17}\)

**Software.** The PPS lab develops and maintains various large software systems, the most important ones being Coq (within the pi.r2 team; Coq is distributed by INRIA) and Ocsigen (developed within the IRILL structure and for which a start-up is currently being created, see Section 1.2). PPS develops also smaller systems, often of a more experimental nature or with more specific applications, or which are parts of larger open source distributions. As explained in Sections 2 and 4, these software developments are an essential part of our research activities. See Section 5 for a complete list.

**Involvement in projects and cooperations.** The PPS lab is involved — or has been involved during the considered period — in many research projects, listed in Section 6: 26 national projects (of which 18 are ANR projects), 11 European projects (of which 4 are EU research projects) and 10 extra-european projects. This certainly demonstrates the good integration of the PPS lab in the international research community and the relevance of our research themes. But this means also that our research activities are extremely dependent on this short-term and goal-oriented funding scheme. We see this as a serious threat for the future.

---

17Remember that conferences are often more selective than journals in Computer Science.
Connection with society. The PPS lab made efforts for explaining its research activities to a larger audience: we have been regularly participating to the Fête de la Science and Di Cosmo has been invited twice to France Culture radio programs. Di Cosmo and Treinen have written various popularization articles, see Section 5. The PPS lab is very active in the Open Source software community and several of its members are regular contributors to various program developments. In particular, Stefano Zacchioli is currently the Debian\textsuperscript{18} Project Leader.

Curien took part to the redaction of the Rapport Petit which preceded the creation of the CNRS institute INS2I. Di Cosmo has coordinated an academic and industrial working group to produce a Report on the French Software Industry, which has been made available to the public in June 2011.

1.8 Conclusion

The evolution of our funding structure has continued, projects of various kinds providing an ever-growing part of our budget. A large part of the travel and invitations made by members of the PPS lab are therefore funded by projects, the regular PPS funding being used in priority for operations of general interest and for members of the lab who have no current access to project grants. This high number of projects of various kinds has made the management of the lab much more complicated, and has turned the production of the reports and of the numerous indicators required by our administrations into a quite intricate task. New management tools, and in particular a global information system, should be designed to help the lab in these administrative duties.

The production of the PPS lab during the period 2007-2012 consists of a significant number of high level international publications of course, but also of the design, production and maintenance of several software products, most notably Coq and Ocsigen. This effort has been done mainly by Université Paris Diderot Assistant Professors, CNRS and INRIA Researchers, who devoted an important part of their research time to these development activities. This software production, supported and encouraged by the new associate institutions pi.r2 and IRILL, represents an essential part of our scientific activities by now. The PPS lab is currently trying to recruit a Research Engineer able to contribute to the development and maintenance of these software products as a support to this evergrowing aspect of our production.

The scientific themes of the PPS lab have significantly evolved during the period. An important evolution is the increase of the research devoted to type theory, both from a practical viewpoint and from a theoretical one. This research effort is complemented by the maturing of our interface with mainstream mathematics, and the integration of new ideas coming from homotopy theory and higher dimensional algebra. Another important evolution of the PPS lab is the growing interest in formal methods applied to Open Source software systems, to concurrent and network programming, and to modeling biological systems.

2 Theme: Proofs and Types


\textsuperscript{18}See http://www.debian.org, one the most important Linux distributions.
2.1 Lambda-calculus and extensions

An important activity of the PPS lab is to study the formal properties of the symbolic structures appearing in actual programming languages. These structures, like pattern matching or closures (explicit substitutions), are naturally understood as canonical extensions of the lambda-calculus whose metaproperties may be then carefully investigated.

- **Lambda-calculi with patterns.** B. Jay and Kesner introduced the Pure Pattern Calculus (PPC) [JK09] which extends previous calculi with pattern matching, including the ρ-calculus by H. Cirstea and C. Kirchner [vO90, CK01, Jay04, FK07]. One difficult question is to find normalizing strategies for these calculi where patterns are also evaluated. E. Bonelli, Kesner, Lombardi and A. Ríos [BKLR12] introduced a specific call-by-need strategy for PPC and showed that it normalizes\(^{19}\). Balabonski studied the efficient implementation of calculi with dynamic patterns, combining an explicit matching mechanism [Bal10a] with an explicit characterization of optimal sharing in a calculus of weak reduction [Bal10b]. This led him to study more generally the sharing of subterms in the implementation of functional programming languages, using ideas from rewriting, and more specifically a **fully lazy sharing** of programs based on abstract labelled terms [Bal12b, Bal12a].

- **Calculi with explicit substitutions (ES).** ES calculi decompose the usual substitution of higher-order calculi into more atomic steps in order to analyze the compilation schemes of functional languages, typically based on closures. While the λ-calculus does not have critical pairs, ES calculi [ACCL91] suffer from diverging reductions on metaterms (terms with metavariables). However, it is possible to reconcile confluence on metaterms and preservation of β-SN\(^{20}\), as in λlx[r] [KL07a], but at the price of complex reduction systems. Kesner [Kes07, Kes08] introduced an alternative ES formalism with full and safe composition, inducing a concise calculus with all the good properties: confluence, simulation of β-reduction, full composition, preservation of β-SN and SN of typed terms [Kes09].

- **Explicit substitution and distance calculi.** The various ES systems treat the content of a substitution as a non-linear resource, i.e. a box that can be composed with another one by means of commutative rules. A box-free net formalism for λ-terms inspired by Intuitionistic MELL\(^{21}\) has recently been proposed [AG09]. Accattoli and Kesner [AK10] studied a term formalism based on the read-back of these nets [Acc11]. This calculus has many good properties: confluence, full composition and preservation of β-SN. Its reduction admits a modular extension implementing a natural operation on nets [AK12a].

- **Permutative theories for the λ-calculus.** The standard operational semantics of λ-calculus is given by β-reduction and is often extended with other rewriting rules allowing the permutation of constructors. The **permutative λ-calculus** of Accattoli and Kesner [AK12b] generalizes all previous extended λ-calculi by taking the permutations as **equivalences**, and not as **reductions**. It preserves β-SN and is Church-Rosser modulo the equivalences.

\(^{19}\)This research is part of a long-term cooperation involving the PPS lab and the CS Departments of University of Buenos Aires and National University of Quilmes, which contributed recently to the creation of the LIA INFINIS http://www.infinis.org/ directed by Kesner, see Section 1.2. It is also supported by the MoDy project of the région Ile-de-France and by several STIC-Amsud partnerships.

\(^{20}\)Strong normalization.

\(^{21}\)Multiplicative Exponential Linear Logic.
2.2 Subtypes and intersection type for XML data processing

CDuce is a programming language designed to query and process XML data, developed and maintained by Castagna and his group at PPS [BCD+11, BCF07]. The CDuce language is motivated by clear practical interests but also has a significant impact on fundamental research.

- **Polymorphic iterators and subtyping.** Even the simplest XML transformations by iteration of basic functions require very expressive type systems and advanced polymorphism in order for the code to be modular and reusable. In XDuce, Xtatic or CDuce these iterators are hard-coded into the language and specific typing rules are added for them. However, this approach soon shows its limits. Castagna and K. Nguyen (LRI) [CN08] proposed a restricted polymorphic language, expressive enough to write complex iterators and simple enough to type them precisely. This restricted language is designed in order to be embedded in any statically typed host language.

In [FCB08], V. Benzaken, Castagna and A. Frisch defined a natural set-theoretic interpretation of types in order to define a decidable subtyping relation. Extending this to polymorphism was deemed impossible because the natural generalization leads to a subtyping incompatible with parametric polymorphism. [HFC09] proposed a partial solution, and then, Castagna and Xu characterized uniformity using a notion of convex model and obtained a subtyping relation for very rich polymorphic types that has all the desired properties and is EXPTIME-decidable.

- **Development of CDuce.** Besides these theoretical investigations towards a polymorphic version of CDuce, the language has been extended in more data-oriented ways. The pattern matching has been extended by a query language [BCCM08, BCCM07] and the interoperability of CDuce has been improved as follows: integration into Ocsigen, web-services in CDuce integrated to Ocaml, and finally, an interface with XProc — an XML pipeline language recently specified by W3C.

2.3 Rewriting and security

Treinen continued at PPS his work initiated at LSV on using techniques from logic, rewriting, and tree automata for the analysis of the security properties of cryptographic protocols in the presence of algebraic properties which allow the modeling of important properties of cryptographic primitives. Together with S. Kremer and A. Mercier (both LSV) he investigated the automated verification of a particular type of security protocols, called group protocols, in the presence of an eavesdropper, i.e., a passive attacker [KMT08a]. They also investigated the combination of algorithms for deciding static equivalence modulo equational theories [KMT09, KMT12]. Static equivalence is a well-established notion of indistinguishability of sequences of terms which is useful in the symbolic analysis of cryptographic protocols. Together with F. Jacquemard, E. Lozes and J. Villard (all LSV), Treinen investigated the model-checking problem of AπL, a spatial equational logic for the applied π-calculus [JLTV11]. The decision algorithm given in that paper relies on a reduction to the first-order theory of finite trees with several distinguished congruence relations, each given by some equational axioms.

---

22 This research has been supported by the ACI TraLaLA and by the ANR domaines émergents project Codex, and has motivated the creation of the ANR Blanc project Typex, starting in 2012.

23 This research is supported by the National Natural Science Foundation of China project Unranked tree-structured data and semantically defined polymorphic type systems.
2.4 Development of the Coq system

The first public release of CoC (an implementation of Coquand and Huet’s Calculus of Constructions) dates back to 1989. The same year, Coquand and Paulin [CPM90] designed an extension of the Calculus of Constructions with a generalization of algebraic types, called inductive types. The resulting Calculus of Inductive Constructions (CIC) became the foundation for the Coq system. It is a logic powerful enough to formalize all everyday mathematics and, at the same time, an expressive dependently-typed functional programming language, with no effects and no non-terminating functions. The development and maintenance of Coq is performed today by pi.r2 together with the team CPR at CNAM.

- **The main components of Coq.** The verification kernel is based on an interpreter of CIC programs and a module system. It is essentially stable with all modifications tied to the evolution of the underlying CIC formalism. The user language is called Gallina, and includes type inference as well as a complex pattern-matching algorithm, implicit arguments and mathematical notations. The system offers a large set of libraries for arithmetics, lists, etc. The tactics enable to conduct proofs, from the basic inference rules of CIC to advanced automation techniques. Finally, extraction transforms programs (or even computational proofs) of CIC to functional programs in OCaml, Scheme or Haskell.

**Development of Coq at pi.r2.** Since the creation of pi.r2, three versions of Coq have been released: 8.2, 8.3 and 8.4. We describe their contribution briefly.

- **Internal architecture.** The integrated graphical interface of Coq (CoqIDE) has been revised: Gross implemented a new communication model based on process interaction rather than on threading (multiple Coq sessions, robustness on non Unix-compliant OS, etc.). Letouzey finalized this work: CoqIDE and Coq are now separate processes. Letouzey also improved the internal architecture of Coq, isolating “plugins” parts that can now be dynamically loaded and introduced an alternative build infrastructure based on ocamlbuild. This allowed him to propose a fully-automated script for building the Windows versions of Coq. This major reorganization of the internal components of Coq aims at better isolating components and making explicit the interfaces between them. In parallel, Régis-Gianas optimized the hash-consing algorithm used in the kernel.

- **Modules.** Soubiran [ES09] unified module implementations and module types into the single concept of theory equipped with a notion of equivalence providing Coq with a transparent namespace. This primitive notion of theory may be then separated into name-space and structure, thus leading to a new merging-of-structure combinator.

- **Extraction.** Letouzey has improved and maintained the current implementation of the Coq extraction, in particular towards Haskell. Many practical aspects of extraction have been reworked to provide a better user experience, in particular the code transformations done to optimize the extracted programs. Letouzey has also worked on better ways to allow replacing Coq code or axioms by external code during extraction, or ways to fine-tune the parts of Coq code that are kept or not during extraction.

- **Type inference, tactics, unification and type classes.** Sozeau corrected issues with the unification algorithm and enhanced it to support universes. He improved the type-class [SN08] implementation and introduced a new system based on type classes to allow for generalized rewriting by custom relations [Soz09]. To improve the power of induction tactics, Herbelin added new heuristics for second-order pattern-matching and Letouzey extended the pattern-matching feature of the tactic language.
Towards the next generation of proof assistants. Apart from this essential mission of improving and maintaining the Coq proof assistant, one crucial objective of the pi.r² team and the justification of its integration in the PPS lab is to design and experiment the new concepts which will be at the heart of future generations of proof assistants\(^\text{24}\).

- **Libraries.** Letouzey started a deep reform of some parts of the Standard Library of Coq (mainly FSets and Numbers): all number representations come with the same set of basic functions and the same set of lemmas about them\(^\text{25}\). These modernized and certified libraries are also one of the first large-scale experimentations of the new features of modules and type-classes. Sozeau [Soz12] worked with members of the Foundations team at Nijmegen on enhancing the implementation of type-classes to suit the needs of the development of the MathClasses formalization of abstract algebra\(^\text{26}\). This gave rise to an experimental implementation of forward reasoning for instance resolution.

- **Certified extraction.** Glondu [SG09, Glo12] proposed an extraction algorithm for Coq that produces correctness proofs of the generated programs and proved manually the correctness of the extraction of some basic functions involving recursion, logical preconditions, and re-use of previously defined functions. He proved that reductions in the source language can be simulated in the target language (the converse was already known).

- **Proof languages.** Puech worked on an inference mechanism in Coq for the recognition of mathematical structures, and on its integration into the automated theorem proving tactics of Coq, this enhancing their efficiency and expressiveness. Proof assistants like Coq provide a read-eval print loop interface with a global undo system, implemented in an ad-hoc way. Puech and Régis-Gianas [MRG10] developed a principled and general approach to incrementality in typed formal language, by means of a fine-grained analysis of dependencies.

- **Forcing in Coq.** Sozeau, N. Tabareau and G. Jaber studied an extension of Coq with Cohen forcing, formulated in presheaf categories, and adapted to dependent types [JTS12]. Sozeau is developing a plugin for Coq in order to define forcing layers and automatically translate statements in these layers to core Coq, with expected applications to verification of imperative programs and low level code.

2.5 Formalization in Coq and in other proof assistants

Besides its participation in Coq development, the PPS lab is also interested in experimenting and formalizing within various proof assistants.

- **Proofs of programs.** Régis-Gianas and F. Pottier worked on proofs of higher-order programs using Hoare Logic [RGP08]. In a similar spirit, Vouillon has formalized in Coq the semantic argument based on step-indexing establishing the correctness of low level programs in [VM04]. In collaboration with P. Audebaud and C. Paulin-Mohring, Régis-Gianas worked on proofs of probabilistic programs. In another direction, Sarnat and Zeilberger investigated CPS translation and defunctionalization [Rey72] from the viewpoint of their effect on termination proofs — with the practical aim of building a compiler from higher-order proof assistants like Coq to first-order ones like Twelf.

---

\(^{24}\) Part of this research is performed within the ANR Ingénierie Numérique et Sécurité Paral-ITP project.

\(^{25}\) This work has been supported by the ANR Sécurité, Systèmes embarqués et Intelligence Ambiante project CompCert.

\(^{26}\) Part of this research was performed within the European ForMath EU project.
Proofs in logic and type theory. On the meta-theoretic side, Siles has formalized in Coq properties of untyped Pure Type Systems (PTS) possibly formulated as sequent calculus\(^\text{27}\). He has extended this work to PTSs with judgemental equality, and proved that both presentations are equivalent. Ilik formalized in Coq his completeness proof for full intuitionistic logic with respect to a newly introduced notion of model based on a dependently-typed continuation monad. In this way, he avoided the need of the Fan Theorem and of delimited control operators [Ili10b]. A. Bove, A. Krauss and Sozeau [BKS12] have compared various formalization methods for recursive functions available in constructive type theories like Agda or Coq and classical systems like HOL.

2.6 Certification of cost annotating compilers

The formal description and certification of software components is reaching a certain level of maturity with impressive case studies ranging from compilers to kernels of operating systems. A nice illustration is provided by the proof of functional correctness of an optimizing compiler of the Cminor language [Ler09]. A natural refinement would be to integrate the execution cost of the compiled code — and to construct a certified C compiler that would produce the expected low level code together with a cost annotation of the high level code\(^\text{28}\). The state of the art in e.g. Scade (Esterel Tech.) is that the reaction time of the program is estimated by abstract interpretation on the binaries, using specific knowledge of the processor and requiring explicit (and uncertified) annotations of the binaries. In contrast, Amadio, Ayache and Régis-Gianas [AARGS10, ARG12] have designed a general labelling method which enables the transformation of a compiler into a “cost annotating compiler”. In addition, they have provided evidence for the usability of the generated cost annotations by implementing a Frama-C plugin which relies on their cost annotating compiler.

2.7 Dependent types

The design and theory of a new generation of proof assistants relies ultimately on a deeper understanding of their type-theoretic foundations, and possible extensions, typically with pattern-matching and coinductive reasoning. Just as in Section 2.1, these investigations should be performed in an appropriate framework, provided in this case by Pure Type Systems (PTS), which generalizes HOL as well as the original Calculus of Construction.

Unification and pattern-matching in PTSs. The CIC system underlying Coq has a decidable syntax-directed presentation. However, only its typed equality presentation can be shown consistent by semantic means. Herbelin and Siles [SH10, SH12, Sil10] bridged this gap by showing that every PTS is equivalent to its typed counterpart; this opens the way to a study of unification algorithms in Coq. In collaboration with B. Ziliani and A. Nanevski, Sozeau started a project to formalize (on paper) the improved unification algorithm, taking into account its advanced features. This will enable the Coq system to handle the delicate usage patterns developed by G. Gonthier in the INRIA-MSR Mathematical Components team, and to scale the system to large formalizations. In a similar spirit, Herbelin and Boutillier have worked on a new simulation of Agda’s style dependent pattern-matching [Coq93] in Coq which relies on earlier work by Boutillier [Bou12]. Sozeau developed the Equations [Soz10b, Soz10a] plugin to build and reason on dependently-typed programs and applied it to prove the metatheory of LF.

\(^{27}\)See [https://www.lix.polytechnique.fr/~vsiles/coq/formalisation.html](https://www.lix.polytechnique.fr/~vsiles/coq/formalisation.html)

\(^{28}\)This project is the starting point and purpose of the EU Certified Complexity (CerCo) project. This research is also supported by the INRIA ARC Eternal project.
(Co)Inductive Types and PTSs in Sequent Calculus. Herbelin, Sarnat and Siles initiated a sequent-calculus presentation of CIC, starting from a simple type theory with inductive and coinductive types, pattern matching, and guarded least and greatest fixed-points. Accordingly, Herbelin and Siles investigated formulations of PTSs in sequent calculus, building on previous work by Herbelin [Her94] and Lengrand [Len06].

Normalization of CoC and CIC. During his stay in the PPS lab, Abel designed and partially formalized a normalization proof of CoC (Calculus of Constructions) in Coq based on normalization by evaluation [Abe10]. With M. Pagano he studied normalization by evaluation for Martin-Löf type theory with singletons [ACP09]. Using a connection to abstract machines, Boutillier extended to all possible reduction strategies the structural guard condition for fixpoints, whose role is to ensure the termination of Coq programs.

Implicit calculus of constructions. Together with B. Barras, Bernardo developed an Implicit Calculus of Constructions [Miq01] with dependent sums and decidable type inference, introducing its syntax, proving subject reduction and studying its denotational semantics. In discussions with Bernardo and B. Barras, Abel investigated the relationship of proof irrelevance and implicit quantification in CoC [Abe11].

Linear dependent types. One main ambition of the PPS lab in future years is to connect proof-assistants with linear logic and other advanced tools in programming language semantics. Developing ideas in his PhD thesis [Spi11], Spiwack started investigating linear dependent types in order to understand what kind of set theory based on setoid arises in this setting. In particular, Spiwack observed that setoids give rise to a quasitopos inside Coq.

2.8 Classical logic, operational duality and effects

The discovery of polarities and the operational duality between programs and environments led to a finer logical and syntactic account of programming languages, including their evaluation strategies. This general picture is enhanced by the key observation that side effects may be incorporated into intuitionistic proofs by extending the logic with axioms like the Markov principle or the double negation shift.

Polarization and focalization. The Curry-Howard interpretation of the sequent calculus (rather than natural deduction) initiated by Herbelin at the beginning of the 1990s led to the design with Curien of a symmetric calculus [CH00] exhibiting the dualities between programs and environments, and between call-by-name (CBN) and call-by-value (CBV) strategies. Starting from this work, Munch-Maccagnoni [MM09] designed focalizing system L as a term syntax for polarized classical logic and LL based on J.-Y. Girard’s ideas [Gir91]. Curien and Munch [CMM10] investigated in this framework the shift from a focalized syntax of cut-free classical proofs to a confluent calculus of focalized and polarized classical proofs.

Axiomatization of CBV and call-by-need. Herbelin and Zimmermann designed an original reduction semantics of CBV λ-calculus that is both complete and confluent [HZ09]. Z. Ariola, Herbelin and Saurin formulated the call-by-need strategy in the spirit of duality, leading to a call-by-need λμ-calculus and unveiling at the same time a new and intriguing evaluation strategy [AHS11] dual to the usual call-by-need. In parallel, Munch-Maccagnoni investigated a notion dual to the thunks of CBV λ-calculus which allow to simulate CBV in CBN.

Effects and proofs. Herbelin discovered that, in an intuitionistic logic extended with a very restricted classical principle and a control delimiter, Markov’s principle becomes provable [Her10]
and this incidentally provides with a syntactic proof that adding Markov’s principle to intuitionistic logic retains the existence property. Similarly, Ilik designed an intuitionistic proof system integrating the double negation shift [Ili10a]. Herbelin investigated extensions of these two systems to other kinds of effects, and in particular, designed a logical formalism with memory management that allows to prove in direct-style any statement provable using the forcing method, starting from ideas by Krivine [Kri08]. Despite the use of effects, the whole framework satisfies the disjunction and existence property. Two typical applications are global-memory proofs of the axiom of countable choice and an enumeration-free proof of Gödel’s completeness theorem.

- Kripke semantics for classical logic. Herbelin and G. Lee gave a simple proof of Kripke completeness for the negative fragment of intuitionistic logic [HL09]. Herbelin, Ilik, and G. Lee also defined a Kripke semantics of classical logic and proved completeness constructively [ILH10]. These two works were formalized in Coq [HL12] and provided an interesting case study on the formal representation and manipulation of binders. Herbelin and Ilik used their systems with side effects in order to prove constructively the completeness of intuitionistic logic wrt. Kripke semantics in the presence of positive connectives (disjunction, existential quantification).

2.9 Delimited control

Delimited control (DC) is an advanced control mechanism which enables the programmer to insert prompts inside programs, and to consider them as the new root of the environment during evaluation. This mechanism enables in particular to reify and reflect any effect (references, exceptions, states, dynamic bindings...) which can be defined as a monad in the system, as first observed by Filinski [Fil94].

- Delimited control and $\Lambda\mu$. The $\Lambda\mu$-calculus was originally introduced by Saurin [Sau05] as a variant of Parigot’s $\lambda\mu$-calculus [Par92] enjoying the Böhm separation theorem. Herbelin showed with S. Ghilezan [HG08] that the $\Lambda\mu$-calculus provides a CBN version of Danvy-Filinski’s CBV calculus of DC. Much work has been devoted to the precise description of the connection between the various calculi and the operational properties of the $\Lambda\mu$-calculus [HS09, Sau08, Sau10b, HG08, Sau10a].

- Logical interpretation of delimited control. Herbelin and Ilik discovered a connection between DC and the double-negation shift schema which is the main ingredient in providing an interpretation of the double-negation translation of the Axiom of Choice, and hence of consistency of Analysis [Ili10b]. Zeilberger studied in [Zei10] a generalization of the classical interpretation of polarities, with the aim of better understanding DC, and of building a stronger connection between classical polarities and intuitionistic effect systems such as Levy’s call-by-push-value [Lev99]. Munch-Maccagnoni [MM11] investigated DC operators and delimited CPS translations from the point of view of LL, and provided a polarized DC calculus with two dual modes of evaluation for expressions (strict and lazy).

2.10 Tensorial proof-nets and combinatorics of proofs

The operational duality between programs and environments may be alternatively studied in syntactic frameworks inspired from LL proof-nets and nicely connected to interactive models of computation like game semantics. This idea leads to a new generation of proof-nets with different

---

29 This research takes place in particular within the Semacode INRIA associate team.
degrees of sequentiality, with L-nets going from the purely sequential tensorial proof-nets to the familiar concurrent LL proof-nets. These combinatorial representations open the general problem of enumerating the proofs of a given type, or at least of deciding when it is inhabited in an effective fashion.

- **Tensorial logic.** Despite their shared origins, LL and game semantics have slowly diverged and become schismatic in the late 1990s\(^{30}\), driven by earlier influential works by Lorenzen \([\cdot]\), Berry and Curien \([\cdot]\) and Gandy \([\cdot]\), and by the need of developing new models of non functional features (references in particular). The situation is confusing and problematic if one wishes to formulate a clean algebraic theory of programming languages. In order to resolve the gap, Melliès introduced tensorial logic \([MT10]\), a primitive logic of tensor and negation, which refines LL by relaxing the hypothesis that negation is involutive. Somewhat unexpectedly, this slight change in the logic leads to a notion of tensorial proof-net which unifies game semantics and LL proof-nets. This work also reveals that game semantics is a diagramatic syntax of linear continuations, formulated in the 2-categorical language of string diagrams, and nicely connected to functorial knot theory. It provides a satisfactory equational theory of proofs, since two proofs are equal precisely when their proof-nets are equal modulo topological deformation, resolving the traditional problem of equality with multiplicative units in proof-nets in LL.

- **Sequentialization.** Faggian and P. Di Giamberardino applied ideas coming from parallel and sequential strategies as expressed in L-nets \([CF12]\) to the syntax of proof-nets, producing a new, elegant and compact proof of sequentialization \([DGF08]\)\(^{31}\).

- **Ticket entailment.** Padovani \([Pad10]\) worked for several years on, and eventually solved, the difficult question of the decidability of the Logic of Ticket Entailment (problem #2 in the TLCA list of open problems), raised by Anderson and Belnap in 1960. The question is to decide whether a given type in this “non-commutative” and relevant \(\lambda\)-calculus is inhabited. Padovani showed the problem to be decidable by applying a series of clever combinatorial transformations on proofs, and by establishing that the process terminates using an intricate Kruskal argument. The paper was accepted for publication in December 2011.

### 2.11 Resource control in higher order languages

Building on fragments of LL, a number of typed functional languages have been designed defining exactly the functions computable in, say, polynomial time. Since usual type systems are too poor to account for the behaviour of programs with effects, an important research issue is to integrate effects into these resource-bounded languages. This is typically achieved by abstracting the state space into regions following the type and effect discipline advocated by Lucassen and Gifford \([LG88]\).

- **Complexity and execution time in LL.** D. De Carvalho, L. Tortora de Falco and Pagani generalized to LL earlier results by De Carvalho on the interpretation of the relational semantics in terms of computation time \([DCPTDF08]\) with strong connections to the Taylor expansion of DiLL\(^{32}\). Dal Lago worked with P. Baillot and P. Coppola on optimal reduction \([BCDL07]\) and quantum computations from a complexity point of view, and with S. Martini on the problem of finding good measures of execution time in the \(\lambda\)-calculus \([LM08]\).

---

\(^{30}\)This research has been supported by the ANR Blanc project CHOCO.

\(^{31}\)This research is supported by the ANR Blanc project Logoi.

\(^{32}\)This research was supported by the MIUR project Réseau italo-français de recherche en logique et géométrie du calcul.
Reducibility candidates and termination. Amadio [Ama09b] proved using reducibility candidates that a suitable stratification of the type and effect system entails termination of the typable programs. The proof technique covers a simply typed, multi-threaded, call-by-value \( \lambda \)-calculus, equipped with a variety of scheduling (preemptive, cooperative) and interaction mechanisms (references, channels, signals) as well as a fragment of the \( \pi \)-calculus [Ama11].

Regions and determinacy. Building on this approach, Amadio and Madet [ABM10] designed an affine-intuitionistic system of types and effects which can be regarded as an extension of Barber-Plotkin Dual Intuitionistic LL to multi-threaded programs with effects. They introduced a discipline of region usage that entails the confluence (and hence determinacy, cf. section 4.2) of the typable programs.

Types for elementary and polynomial time. Amadio and Madet [MA11] developed a type system which corresponds to elementary time. They provided a new combinatorial proof of termination in elementary time, developed an extension of the approach to a CBV \( \lambda \)-calculus with multithreading and side effects and introduced an elementary affine type system that guarantees the standard subject reduction and progress properties. The typing system is sufficiently flexible to allow the programming of iterative functions with side effects. Madet developed a polytime variant of these results.

2.12 Resources in differential linear logic

DiLL is an extension of LL, introduced by Ehrhard and L. Regnier [ER06], in which the structural rules of the exponentials (weakening, dereliction and contraction) have exactly dual counterparts: costructural rules have a strong algebraic flavour. Coweakening is related to the presence of a zero proof object, cocontraction is related to the possibility of adding proof objects and coderelection corresponds to differentiation wrt. a given hypothesis. DiLL provides new mathematical foundations to resource \( \lambda \)-calculi introduced by Boudol [BL96]. DiLL-based \( \lambda \)-calculi have two kinds of application constructions: the ordinary one and an application which represents differentiation. Terms without ordinary application have strong normalization properties, and all terms can be represented as infinite sums in this restricted fragment (Taylor expansion).

DiLL and processes. Ehrhard and Laurent have shown how DiLL can be used to represent concurrent processes by encoding a version of the \( \pi \)-calculus in an untyped DiLL. They proved that the corresponding translation is a bisimulation. They also proposed a simpler and more direct encoding of the calculus of solos [EL10a]. Starting from the graphical intuitions behind these translations, Ehrhard developed with J. Ying a version of CSS which deals with trees instead of words; in this setting, parallel composition has a richer, graphical, structure. Gimenez [Gim09] studied a generalization of the promotion rule of LL, completing the symmetrization of the LL exponentials initiated with DiLL. Deep inference (see Section 2.13) seems to provide a convenient setting to express this rule.

DiLL and exceptions. Munch–Maccagnoni developed a symmetric syntax for DiLL and gave a logical interpretation of exceptions using DiLL co-dereliction to model the fact that an exception binder can catch only one exception. Then, Herbelin’s implementation of Markov’s principle corresponds to the validity of co-dereliction on recursively positive formulae.

---

33 This research is part of the ANR Complice project.
34 This research was supported by the ANR Jeunes chercheurs, jeunes chercheuses No-CoST.
35 This research is funded by the French-China ANR project LOCALI.
**DiLL, \(\lambda\)-calculus and nets.** Ehrhard investigated the connection between Taylor expansion and normalization of \(\lambda\)-terms by introducing a finiteness structure (in the sense of [Ehr05]) on finite resource terms whose definition uses differential \(\beta\)-reduction, showing that the Taylor expansion of terms typeable in system F are finitary in that sense [Ehr10]. With Carraro and A. Salibra, he designed a differential extension of combinatory algebras, providing a purely algebraic presentation of resource calculi [CES10]. Pagani [Pag11] discovered a correctness criterion for DiLL proof structures whose relational semantics is finitary in the sense of Ehrhard’s finiteness spaces [Ehr05]. Pagani and Tasson [PT09a] characterized the DiLL nets which appear in the Taylor expansion of LL nets. Giminez proved [Pag09] combinatorially cut-elimination for simply typed DiLL nets and [Gim11] provided an alternative proof, based on reducibility and therefore extendable to higher order LL. Pagani and Tranquilli [PT09b] proved standardization and confluence for the resource calculus based on DiLL.

### 2.13 Deep inference

Deep inference is a formulation of proof theory where every proof is obtained as a series of symbolic transformations from the hypothesis to the conclusion. This approach to proof theory is developed by Parigot in the PPS lab and takes place within a rich European network of cooperations.\(^{36}\)

**Quasipolynomial normalization in deep inference.** All known cut elimination procedures for usual proof systems for classical propositional logic are exponential and there were good reasons to think that this could not be improved. However, Jeřábek [Jer08] showed in 2008 that cuts in propositional-logic deep-inference proofs can be eliminated in quasipolynomial time. Together with P. Bruscoli, A. Guglielmi and T. Gundersen, Parigot [BGGP10] gave a direct proof of this result, using atomic flows.

**Proof systems.** In the sequent calculus, only very limited means of combining proofs are available. Together with A. Guglielmi and T. Gundersen, Parigot [GGP10] devised a very simple and natural logic-independent proof calculus, Open Deduction, where proofs can be freely composed by connectives, and proved its basic properties. Open Deduction allows to increase the degree of parallelism in proofs. With Gundersen and W. Heijltjes, Parigot devised a new term calculus for natural deduction inspired from deep inference. Building on the new possibilities given by Open Deduction, Parigot devised a symmetric natural deduction system. The key point is the distinction between connectives and metaconnectives which reflect them at the level of proofs.

### 3 Theme: Mathematical Foundations of Programming

**Participants (permanents).** Berline, Bucciarelli, Curien, Ehrhard, Faggian, Gaucher, Guiraud, Harmer, Herbelin, Joly, Krivine (Jean-Louis), Légrangërard, Melliès, Métayer, Miquel, Rozière, Tasson, Varacca.

\(^{36}\)This research is supported by the PHC PAI Germaine de Staël Deep Inference and the Essence of Proofs, by the PHC Nouveaux systèmes de déduction et méthodes de normalisation pour l’inférence profonde (with UK) pursued with the PHC Outils logiques d’analyse des programmes and the ANR-FWF project Structural, and with the CNRS-GNSF project Structural and Computational Properties of Logics (with Georgia). It has also been supported by the ANR Blanc project Infer.
3.1 Game semantics and interactive models of computation

Games provide a mathematical interpretation of programs as strategies where moves are elementary data tokens. This interpretation extends LL and other logical systems, suggesting often refinement of these syntaxes.

- **Algebraic account of arena games.** Harmer, M. Hyland and Melliès [HHM07] developed an algebraic and combinatorial account of arena games and innocent strategies, inspired by simplicial sets in algebraic topology. The reconstruction of innocent strategies relies on the discovery of a distributive law \( \lambda : \mathcal{C} \rightarrow \mathcal{D} \) between a monad \( \mathcal{C} \) and a comonad \( \mathcal{D} \) living in the category of alternating games and sequential strategies. An innocent strategy between the games \( A \) and \( B \) is then recovered as a sequential strategy \( \lambda \cdot A \rightarrow \lambda \cdot B \). In that way, the subtle combinatorics of cut-elimination between innocent strategies (and thus derivation trees) is entirely handled by the distributivity law \( \lambda \), and the traditional category of arena games and innocent strategies is recovered as its Kleisli category.

- **Dialogue categories.** The notion of dialogue category plays the same role for tensorial logic as the notion of cartesian closed category plays for the simply-typed \( \lambda \)-calculus. Accordingly, the shift from LL to tensorial logic comes together with a shift from \( * \)-autonomous categories to dialogue categories. A natural question is thus to understand which results obtained for \( * \)-autonomous categories may be lifted to dialogue categories. Melliès has developed a comprehensive study of dialogue categories with this question in mind. An interesting outcome has been the discovery of a microcosm principle in proof theory generalizing a similar microcosm principle in higher dimensional algebra. The idea is that logical negation is a duality at dimension 1 which relies on the involutive duality at dimension 2 between a category \( \mathcal{C} \) and its opposite category \( \mathcal{C}^{op} \). These two levels of duality are typically confused in LL, this forcing the logical negation of LL to be involutive. From this microcosm principle, Melliès deduces the surprising fact that the familiar decomposition of implication \( A \Rightarrow B = A^* \lor B \) in LL already holds in cartesian closed categories, that is, in a purely intuitionistic setting. The formula requires however to decorrelate the category \( \mathcal{C} \) from its opposite category \( \mathcal{C}^{op} \) and to work in what Melliès calls a (cartesian closed) chirality.

- **Abstract machines for dialogue games.** Curien and Herbelin [CH09b] developed the paradigm of computation as interaction in the framework of abstract Böhm trees, introduced by Curien in earlier work. The approach can be understood as a type-free, operationally-oriented view of game semantics. The main benefit of the generalization is that it offers the right level of generality for explaining the mechanism of computation at hand in the \( \lambda \)-calculus and similar sequential languages. The paper presents syntactic support for this claim, as well as numerous examples (like CBV) illustrating the generality of the underlying computing device.

- **Fixpoints and Totality in Game Semantics.** During his PhD thesis, Clairambault worked on the game-theoretic aspects of the normalization problems encountered in typed \( \lambda \)-calculi. With Harmer [CH09a], he gave abstract conditions on strategies ensuring the finiteness of their interactions and as a consequence preservation of totality by composition. The tools introduced to that effect were later strengthened [Cla11] to give a complexity result, which provides bounds to the length of linear head reduction sequences on simply-typed \( \lambda \)-terms — such bounds were only known for standard beta-reduction. Clairambault [Cla09] also investigated the issue of totality in the presence of infinite data types such as inductive and coinductive datatypes.
3.2 Games, Ludics and Concurrency

Faggian investigated a game-semantic model of parallel computation based on Ludics, on which she has been working in collaboration with Curien [CF12] and M. Piccolo [FP07a, FP07b, FP09]. The results lead to a better understanding of the relationship between parallel and sequential strategies, and of the relationship with models of True Concurrency.\(^{37}\)

- **A parallel model of computation: graph strategies.** Starting with the pioneering paper by S. Abramsky and Melliès [AM99], several proposals have emerged — with different motivations — towards a notions of strategy where sequentiality is relaxed to capture a more asynchronous form of interaction. Faggian and Curien [CF12] relate parallel and sequential strategies and sequential strategies by showing how strategies represented by graphs, with partial ordering information, can be associated to a set of tree-like strategies, and how conversely, sequential strategies can be relaxed into more asynchronous ones.

- **Ludics, true concurrency and π-calculus.** Faggian and Piccolo [FP07a, FP09] investigated a Game Semantics where strategies are partial orders (called po strategies) and composition is a generalization of merging of order. The bridge between game semantics and concurrency theory is also the basis for a game semantical analysis of the linear π-calculus, introduced by N. Yoshida, K. Honda and M. Berger (2001), and whose typing is based on LL\(^{37}\). Faggian and Piccolo [FP07b] established an exact correspondence between process calculi features and game-semantic notions, by showing that ludics is a model for the finitary linear π-calculus.

- **Ludics with repetitions.** Faggian and Basaldella [FB11, FB09] provided an extension of ludics which allows repetitions (exponentials). The solution proposed here consists in using non-uniform (non-deterministic) tests, which are obtained by using ideas developed in [FP09].

3.3 Vectorial models of differential linear logic

Investigations on denotational models of LL where formulae are interpreted as topological vector spaces led in the early 2000s to the introduction of differential LL and to a new understanding of resource λ-calculi.

- **Lefschetz spaces and Convenient spaces.** Starting from [Ehr05], Tasson [Tas09b] extended to more general linearly topologized vector spaces\(^{38}\) the constructions required to interpret LL. The objective was to obtain a basis-independent, and therefore intrinsic and more geometrical, presentation of these constructions. Related to this work is the joint work on convenient spaces carried out by Richard Blute, Thomas Ehrhard and Christine Tasson [BET10]. They built a model of differential linear logic using convenient spaces, that is locally convex spaces that enjoy the same kind of duality.

- **Transport.** In the finiteness spaces model, fixed points cannot be interpreted [Ehr05, Vau]. Indeed, this semantics focusses on termination. Yet, some algebraic types such as integers, finitely branching trees can be built as fixed points and are finitary (that is they can be modeled using finiteness spaces). Tasson and L. Vaux [TV10] studied the constructions that allow the interpretation of such types. They devised a new transport method to build new finiteness spaces and applied it to the construction of algebraic types in finiteness spaces.

\(^{37}\)This research is supported by the ANR blanc project Logoi.

\(^{38}\)A notion of topological vector spaces introduced by Lefschetz in 1943 which differs radically from the usual ones since the field is discrete and the topology is generated by subspaces.
Categorical models of DiLL and of resource $\lambda$-calculi. Following the work of R. Blute, R. Cockett and R.A.G. Seely on differentiation in cartesian categories, Bucciarelli, Ehrhard and Manzonetto have designed a general notion of cartesian closed categories with differentiation [BEM10]. In a survey paper currently under revision, Ehrhard presents a categorical semantics of DiLL in the framework of linear logic models, in the line of the original presentation of DiLL in [ER06] and of the work of Fiore [Fio07]. He also introduces a surprisingly simple categorical axiomatization for an operation of antiderivatives which is based on the standard Poincaré’s Lemma of analysis.

3.4 The structure of exponentials

The exponentials are unary connectives, or modalities, of LL which allow for the duplication of proofs (data) during cut-elimination (execution). They are thus essential to understand the complexity of computation, and their fine-grained analysis is crucial.

- **Exponentials as limits of tensor powers** The exponential modality of LL transports every formula $A$ to a commutative comonoid $!A$ which may be duplicated in the course of reasoning. Taking inspiration from the construction of the symmetric algebra $SA$ of a $k$-module, Melliès, Tabareau and Tasson [MTT09, MTT10] introduced a general recipe to compute the free exponential modality $!A$ as a sequential limit of symmetrized tensor powers $A^{\leq n}$, where each space $A^{\leq n}$ represents $k \leq n$ symmetric copies of the original space $A$. The idea is then to “glue” appropriately each space $A^{\leq n}$ in the next space $A^{\leq n+1}$ and to compute the limit. The formula works in any symmetric monoidal category where the sequential limit of the $A^{\leq n}$ exists and commutes with the tensor product. This basic recipe provides an unified account of apparently different constructions dispatched in the literature, either on coherence spaces, on sequential games, or on topological models. It also reveals an unexpected limitation of recursive types: indeed, the sequential limit construction of $!A$ cannot be expressed using a recursive type, and requires instead a series of advanced operadic ideas developed for that purpose by Melliès and Tabareau [MT08].

- **Exponentials with infinite multiplicities.** In the relational model of LL, the Taylor formula holds semantically. For that reason, any model of the pure $\lambda$-calculus in this model is sensible. To encompass this limited equational expressivity, Carraro, Ehrhard and Salibra [ECS10] have introduced new exponentials in the relational model, in which multiplicities can be infinite (in various different ways). The availability of so many different exponentials was a surprise, and Bulteau’s PhD thesis aims at understanding their general structure.

- **Extensional collapse of the relational model.** The CCC associated with the relational model of LL is not well-pointed. It seemed therefore natural to try to characterize its “extensional collapse” (of course this notion has to be carefully defined beforehand). Ehrhard proved that this collapse coincides with a CCC of Scott domains (prime-algebraic complete lattices actually) and Scott-continuous functions. This result is obtained by means of a new linear duality on preordered sets, leading to a new model of LL which contains models of the pure $\lambda$-calculus [Ehr12b]. This construction can be used to reduce to purely combinatorial arguments normalization results which are usually proved by reducibility [Ehr12a]. Intuitively, all the logical complexity of reducibility is encapsulated in the model.

3.5 Models of the pure $\lambda$-calculus

The pure $\lambda$-calculus is a Turing-complete model of computations with deep connections to proof systems and to algebra and categories. Differential LL provides a new and refined approach to the
various aspects of the $\lambda$-calculus.

- **Lambda-calculi with resources and tests.** The differential $\lambda$-calculus of Ehrhard and L. Regnier and its syntactic variations mostly due to Tranquilli, referred to as resource lambda-calculi, have been extensively studied in recent years. The pure resource calculus may be interpreted in the simple reflexive object of the category of sets and relations described in [BEM07]. Motivated by the study of this interpretation, Bucciarelli, Carraro, Ehrhard and Manzonetto [EBCM11] devised an extension of the pure resource calculus by the new syntactic category of tests, that can be thought of as a basic exception mechanism. The relational model is proven to be fully abstract for this extension. Then, via a syntactic operation of test-expansion, it is shown that full abstraction holds also for the promotion-free resource calculus without tests.

- **Expressive power of tests.** Somewhat surprisingly, Breuvart [Bre12] showed that the full abstraction property established in [EBCM11] does not extend to the resource calculus without tests. He also proved that the pure $\lambda$-calculus extended with tests is powerful enough to define all the (compact) elements of Scott’s $D_\infty$ model. The full abstraction theorem for the pure $\lambda$-calculus with tests resulting from this definability property implies, by test elimination, the full abstraction of $D_\infty$ for the pure lambda-calculus. This provides a new proof of a classical result and a new approach for attacking open problems of the same family.

- **Model theory of the pure $\lambda$-calculus.** Berline, Manzonetto and A. Salibra [BMS07] introduced a notion of effective $\lambda$-model, which covers in particular all the models individually introduced in the literature and is just strong enough to force the interpretation of a normal term to be decidable. They proved that the order theory of an effective model is never r.e. It follows that its equational theory cannot be $\lambda_3$ nor $\lambda_3\eta$. In [BS08], Bucciarelli and A. Salibra showed that the class of graph models possesses a minimal theory. Recently, together with Carraro [CSB12], they generalized this result to any class of models having the finite intersection property and the ultraproducts property. Many natural classes of models have these properties.

### 3.6 Domain theoretic accounts of randomness and non determinism

Modeling probabilistic computations requires extending domains with stochastic features. The standard approach consists in endowing the model with a stochastic monad but other approaches are possible, such as the notion of probabilistic coherence spaces.

- **Probabilistic coherence spaces.** Danos and Ehrhard [ED11] refined the notion of probabilistic coherence spaces [Gir04] and extended the interpretation to the whole of LL. They explored the properties of the associated CCC showing that it is a model of a probabilistic extension of PCF which enjoys a probabilistic version of adequacy. This CCC contains many models of the pure $\lambda$-calculus, including a probabilistic version of Scott’s $D_\infty$. Ehrhard, M. Pagani and Tasson have proven a probabilistic adequacy of this model wrt. a probabilistic extension of the pure $\lambda$-calculus [EPT11].

- **Random variables and domains.** Varacca and J. Goubault-Larrecq developed a domain-theoretic description of random variables [GLV11]. The domain-theoretic version of measures are called continuous valuations and form a monad in the category of continuous domains; such monads are notoriously difficult to construct.

- **Monad-free semantics of non determinism.** Vaux’s [Vau09] algebraic $\lambda$-calculus extends ordinary typed and untyped $\lambda$-calculi by the possibility of forming linear combinations of terms of the same

---

39 This research was supported by the CNRS PEPS QuAND project.
type and can nicely be understood as an algebraization of non-determinism in a functional setting. Such extensions of logic are used to prove separation as in [FB11, FB09] or in the work of Breuvart.

Introducing a notion of totality on finiteness spaces — which is simply a closed affine subspace of the space interpreting the corresponding type —, Tasson [Tas09a] built a model of barycentric λ-calculus, a natural restriction of Vaux’s calculus. She proved a definability result for first order types (with P. Hyvernat). Bucciarelli, Manzonetto and Ehrhard provided a sound interpretation of a non-deterministic extension of the λ-calculus in the relational model, and showed that a term has a non-empty interpretation if and only if its head reduction converges [BEM09].

3.7 Realizability and models of ZF set theory

Jean-Louis Krivine has pursued his efforts in developing a theory of classical realizability, mainly in the logical language of Set Theory.

► *Realizing various forms of choice.* In earlier work [Kri03], J.-L. Krivine showed how the Axiom of Countable Choice could be realized using instructions which can be understood as implementing a clock. In [Kri08] he extended this work to a stronger form of choice, relating this properties with instructions which read and write in a global heap. This work needed an axiomatization of the structures required in his classical realizability, leading to the concept of Realizability Algebra. In [Kri11], J.-L. Krivine has extended this approach to Set Theory, for realizing a stronger choice principle. This work involves the construction of a model nicely combining classical realizability and forcing, on top of a classical realizability model. Munch studies a polarized version of realizability algebras with applications to normalization proofs. Herbelin investigated the computational content of the axiom of dependent choice in the presence of classical logic and expressed it in terms of call-by-need evaluation of a coinductive representation of universal quantification over natural numbers [Her12].

► *New models of ZF set theory.* Going in the opposite direction, J.-L. Krivine has also shown how to build classical realizability models in which the real line is “very far” from being well-orderable [Kri12b]. Such results seem out of reach of the standard forcing technique. More recently, J.-L. Krivine has considered models induced by realizability algebras coming from stable models and is exploring their properties.

► *Interactive realizability.* Programs extracted from classical proofs use control operators as a way to implement backtracking and processes of intelligent learning by trial and error. Such programs are often hard to write, difficult to understand and are very inefficient. To improve them, it is necessary to describe exactly what the programs learn and how this knowledge varies during the execution. A first step towards this goal is the theory of Interactive Realizability, based on a notion of state. Aschieri [Asc12] extended this semantics and discovered a new state passing style transformation. He is investigating a relation with the forcing semantics: it seems that his transformation is a very direct, new constructive formulation of forcing.

► *The computational contents of completeness proofs.* It is known from works on normalization by evaluation that completeness proofs for intuitionistic logic along models such as Beth models are basically tools to map proofs of the meta-language into proofs of an object language. This scales to classical logic if one considers models such as boolean algebras. However, the computational content of completeness for classical logic with respect to the truth-values model is of a different

40This research is supported by the ANR Blanc project Récré.
nature. Krivine started to investigate this topic in 1996. Herbelin and Ilik carried on this work and were able to extract a quite simple program which, when applied to a proof of validity of a first-order formula, produces a derivation of this formula.

**Valid formulas, strategies and network protocols.** Legrandgérard and J.-L. Krivine associated a game with any formula of the predicate calculus, for which a winning strategy exists iff the formula is valid, and outlined an interpretation of these strategies in terms of network protocols, showing in particular that these protocols include some basic ones (*e.g.* acknowledgments) and can be composed [KL07b].

**Sound axiomatization of Map Theory.** Map theory is a logical foundation of mathematics which replaces ZF set theory by the pure λ-calculus augmented with Hilbert’s epsilon operator. Berline and K. Grue introduce in [BG12] a substantially simplified axiomatization of Map Theory called $MT$ and prove its consistency under the assumption that there exists an inaccessible ordinal. This is achieved by extending the minimal canonical model of a previous version $MT_0$ of Map Theory [BG97]. This establishes $MT$ as a reliable axiomatization of Map Theory.

### 3.8 Low level languages

An important question of a theoretical and practical nature is to extend the existing typing techniques from high level to low level languages. More specifically, one would like to establish that every the semantic properties of a high level program are preserved by compilation into machine code. To that purpose, A. Appel and D. McAllester have introduced in [AM01] a step-indexing technique where every type is interpreted as a set of machine code programs, each of them required to behave properly for a given number of computational steps. Together with A. Appel and his PhD student C. Richards, Melliès and Vouillon have provided clean semantic foundations to the model by understanding it as a Kripke model defined by a recursive equation between worlds and types. One main insight of this work is that the soundness property of the model (and thus the safety of the well-typed machine code) follows from the Gödel-Lob law satisfied by the necessity modality (called later) of the Kripke model. This idea has become mainstream in the field of low level language semantics [Ahm06, DAB09, BMSS11], and has been recently adapted by G. Jaber, Sozeau and N. Tabareau [JTS12] in order to define recursive types in Coq through a forcing layer.

### 3.9 Operadic tools in semantics

**Monads for computational effects.** An important foundational question is to understand how proof theory (and more specifically linear logic) should integrate computational effects in a properly defined mathematical framework. A key ingredient is provided by the algebraic presentation by generators and relations of computational monads, typically for the state monad, as developed in [PP02]. This algebraic formulation is traditionally based on the assumption that the underlying monad is finitary in the category of sets. In order to adapt the notion of finitary monad to more sophisticated situations, typically the sheaf categories used to describe the local state monad, Melliès has advocated in [Mel12, BMW12] the use of *monads with arities* originally introduced by M. Weber in the field of higher dimensional algebra. Melliès introduces there the notion of algebraic theory in any category equipped with a notion of arities. The notion of algebraic theory with arities is then further investigated and applied to groupoids in collaboration with C. Berger and M. Weber [BMW12]. One main reason for introducing that framework is that the local store monad considered in [PP02] is a monad with arities in the category of presheaves $[\text{Inj, Set}]$ equipped
with the finite sums of representables as arities. This notion of arities leads to the definition of 
graded algebraic theory, developed by Melliès, and to a graded algebraic presentation of local stores 
which provides a higher algebraic foundation to the pioneering work of Plotkin and Power [PP02].

◆ Free models of operadic theories. Melliès and Tabareau [MT08] developed a generic method to 
compute the free model $SA$ of an operadic theory generated by an object $A$ in a given monoidal 
category. The construction proceeds in two stages. First, one computes a coend formula providing 
a specific left Kan extension in the 2-category of categories, functors and natural transformations. 
Then, one shows that the formula defines in fact a left Kan extension in the 2-category of $T$-
categories, $T$-functors and natural $T$-transformations, where $T$ is a 2-monad capturing a specific 
notion of operadic theory: genuine, symmetric, braided, algebraic. The construction was applied 
by Melliès, Tabareau and Tasson [MTT09, MTT10] in order to compute the free commutative 
comonoid $!A$ generated by an object $A$ in various models of linear or tensorial logic.

◆ Operads and distributive laws. From 2006 onwards, Curien became increasingly interested in 
the theory of operads and its generalizations. In [Cur12], inspired by conversations with M. Hyland, 
he shows how to decompose the notion of operad by means of two monads and a distributive law 
between them. Varying the monads, one gets known variants of the notion of operad, and one may 
suggest new ones.

◆ Substitutions and isomorphisms. Curien worked with M. Hofmann (Ludwig-Maximilians-
Universität München) and R. Garner (MacQuarie University, Sydney) on comparing two categor-
ical interpretations of Martin-Löf type theory both overcoming the following mismatch: syntax 
has exact substitutions, while categorical interpretations implements substitutions only up to isos. 
One can strictify the model [Hof94] or change the syntax by introducing explicit substitutions and 
type coercions [Cur93]. They showed that these approaches are related by an adjunction whose 
properties are still under study.

3.10 Homotopy theory and concurrency

Gaucher works on the relations between homotopy theory, in particular categorical homotopy 
theory, and concurrency in computer science. He has introduced several geometric models of concur-
rency, some of them variants of models studied by other authors: multipointed $d$-spaces [Gau09], 
topological locally presentable higher dimensional transition systems [Gau10b] and flows (identity-
free small categories enriched over topological spaces). He has devoted a lot of work to model 
process algebras in this setting, the most challenging operation being of course parallel composition 
with synchronization [Gau08, Gau10a]. This necessitated the introduction of a new kind of 
degeneracy map [Gau10a] to make a non-functorial construction of [Gau08] functorial. A new ho-
motopy theory for higher dimensional transition systems [Gau11] was introduced for that purpose, 
with tight connections with strong bisimulation. In [Gau09], he proved that Goubault’s globular 
CW-complexes are the cellular objects of a combinatorial model category which provides a topo-
logical version of the model category of flows introduced by Gaucher in 2003. This proof adapted 
an idea of Fajstrup and Rosicky. Gaucher’s recent work takes advantage of the most recent discov-
eries in model categories with non-standard class of cofibrations and illustrates the importance of 
topological categories.
3.11 Higher dimensional rewriting and homotopy theory

Higher-dimensional rewriting is a unified and computationally-oriented mathematical setting for various symbolic structures such as abstract, word and term rewriting systems, including first-order functional programs, Turing machines or Petri nets, (in Computer Science) and presentations of monoids, groups and algebraic theories, or braids and knot diagrams with Reidemeister moves (in Mathematics). It was initiated by Burroni [Bur93], followed by Lafont, Guiraud, Malbos, Métayer, Mimram and Rannou, see [Laf07, Mét03, Gui04, Gui06a, Gui06c, Gui06b, LR08, Mé08, BG09, GM09, Mim10, GM10a, GM10b, GM11].

- **Homotopy theory, rewriting and higher categories.** Métayer studies connections between rewriting, homotopy theory and higher categories. The original motivation lies in the following result by Squier [Squ87]: if a monoid $M$ can be presented by a finite, confluent and terminating rewriting system, then its third homology group $H_3(M)$ is of finite type. The key observation is that each rewriting system generates a space of computations naturally endowed with a structure of strict higher category, whence the idea of looking for homological or homotopical invariants of the latter. A major conceptual clarification was obtained by Y. Lafont, K. Worytkiewicz and Métayer in [LMW10], describing a natural (“folk”) model structure on the category of strict higher categories. This new framework subsumes polygraphic resolution [Mé03], and cofibrant objects may be characterized as free higher categories on polygraphs [Mé08]. The homotopical invariance of polygraphic resolutions leads to the notion of polygraphic homology for higher categories shown by Y. Lafont and Métayer [LM09] to coincide with the usual definition in case of monoids.

- **Normalization strategies and polygraphic resolutions.** In a parallel line of work, Guiraud and Malbos interpreted Squier’s construction in higher-dimensional rewriting, extending it from dimension 2 (words) to higher dimensions [GM09]. They gave an explicit way of building higher cells using overlaps of higher rewriting rules [GM10b] based on a mathematical understanding of normalization strategies in every dimension and yielding an algorithmic way to build a polygraphic resolution as defined in [Mé03]. This notion of polygraphic resolution has become a central element of the homotopical study of higher categories: it enables to build explicit resolutions for the formalization of mathematical objects (typically to allow symbolic computations in them) as well as the computation of homotopical and homological invariants, for convergence analysis and classification of normalization strategies.

- **Higher groupoids.** In collaboration with D. Ara (IMJ) Métayer was able to transfer the folk model structure from strict higher categories to strict higher groupoids. They prove in [AMI11] that the resulting model structure on strict higher groupoids coincides with the model structure defined by Brown and Golasinski via crossed complexes [BG89].

4 Theme: Modeling, Analyzing and Conceiving Systems

- **Participants (permanents).** Abbes, Amadio, Balat, Castagna, Chroboczek, Di Cosmo, Harmer, Krivine (Jean), Legrandgérard, Manoury, Ruet, Treinen, Varacca, Vouillon, Zacchioli.

4.1 Models for system biology

- **Foundation of rule-based modeling.** Rule-based modeling, such as with the $\kappa$-calculus, is a graph rewriting formalism with expressive patterns that enable one to describe a wide family of biological...
observables with a single expression. The modeling style of $\kappa$ allows one to describe models of very large systems but it also requires fine analysis tools such as abstract interpretation, refinement techniques and causality analysis that Jean Krivine, Harmer and V. Danos contributed to develop [MDF+10, DFF+10, DFF+09a, DFF+07, DFFK08, DFF+08, DFF+09b]. Curien, V. Danos, Jean Krivine and Zhang studied the expressiveness of $\kappa$ and proved that any set of transformation rules can be simulated by rules each involving at most two nodes [CDKZ08].

To establish a proof of concept of the use of $\kappa$ in modelling real biological systems, Harmer developed a large model (about 300 rules) of the so-called erbB signalling network. Harmer, V. Danos, J. Feret, Jean Krivine and W. Fontana, analyzed the ways in which causal properties of this model impact upon its dynamics [DFFK07].

Sometimes, a modeler might wish to replace a rule by more specific sub-cases. However, in order to do this, we must identify the appropriate collection of non-overlapping sub-cases. This is called a neutral refinement. Harmer, V. Danos, E. Murphy, J. Feret, Jean Krivine and W. Fontana have developed a theory of rule refinement that enables the determination of the neutral refinement of a rule [DFF+08, MDF+10, DFF+10].

The work on rule refinement included the definition of a category of site graphs and embeddings in order to discuss the question of the instances of a rule in a site graph and its symmetries. Harmer, V. Danos and G. Winskel developed a major generalization of the previous categorical formulation in order to define the side-effect-free fragment of $\kappa$ in purely categorical terms [DHW11]. Harmer has developed a meta-language for $\kappa$ in order to allow the direct representation of shared interactions: the modeler describes a hierarchy of agents and writes generic rules that may mention any agent [DFF+09a, Har09].

A $\kappa$-model determines a dynamical system. This system may have many dimensions which can be reduced by performing a static analysis: if a collection of molecules cannot be distinguished by the rules, they may be grouped together. Harmer, J. Feret, V. Danos, Jean Krivine and W. Fontana have developed two distinct methods for performing such coarse-graining [FDK+09, DFF+10, HDF+10].

Tools. All theoretical concepts presented have also been integrated in KaSim, a tool for the simulation and analysis of biological systems modeled in $\kappa$ [DFFK07, KDB09]. Krivine and J. Feret (ENS Ulm/INRIA) are the main developers of this tool which is available under the LGPL license and freely downloadable on GitHub. It has around a hundred downloads at each new release of the binaries, plus the advanced users who regularly download the sources. KaSim was used by the Edinburgh team who won the IGEM best model prize in 2010 and 2011.

4.2 Models for concurrency

Determinacy in synchronous languages. The original model of synchronous reactive programming (SL) [BdS96] assumes that signals are pure in the sense that they carry no values. In this case, the computation is naturally deterministic and bisimulation coincides with trace equivalence.

41 This research is supported by the ANR Bioinformatique project Iceberg.
42 This research was supported by the Programme franco-chinois de recherche avancée project Types, processus et leurs applications en sécurité et systèmes biologiques.
43 http://github.com/jkrivine/KaSim/downloads
44 http://igem.org/Main_Page
45 This research was supported by the ANR Sécurité, Systèmes Embarqués et Intelligence Ambiante ParSec project.
Amadio [Ama07a] has proposed an extension of the SL model where signals carry signal names. The model is called $S\pi$-calculus ($S$ for synchronous). He showed that the notion of bisimulation developed for the (deterministic) SL language [Ama07b] is sufficiently robust to be extended to the $S\pi$-calculus. In the framework of timed CCS, Amadio [Ama09a] provided a characterisation of this bisimulation via the usual labelled transition system. Using this semantic framework, Amadio and Dogguy [AD09] studied determinism and confluence in the $S\pi$-calculus: the two notions turn out to be equivalent. They designed then a decidable type system [AD08] for the $S\pi$-calculus which allows to enforce statically a local confluence property. The type system is based on a notion of affine usage inspired by LL [ABM10].

Dogguy [Dog12] has continued this line of work by developing a type system which enforces confluence and determinism for a timed version of the \(\pi\)-calculus (called Tapis) where interaction is based on channels rather than on signals. Dogguy and Glondu have formalized in the Coq proof assistant the calculus and the type system along with a proof of subject reduction.\footnote{This is a rather challenging development available at http://www.pps.jussieu.fr/~dogguy/~research/tapis.}

- **Handshake Protocol.** The Handshake protocol is an informal discipline for asynchronous communication. It was formalised by VanBerkel [VB93], but Fossati [Fos07] discovered that VanBerkel’s composition of protocols is not associative. Together with Varacca who supervised part of his PhD work, they proposed another model that could account for all Handshake protocols, using a special form of Petri Nets [FV09a]. A second more syntactic model of the protocol was proposed, using a process calculus [FV09b].

- **Biographical reactive systems.** Biographical reactive systems (BRSs) are conceived as a unifying framework for designing models of concurrent and mobile systems. These reactive systems are construed as a set of rewriting rules together with an initial bigraph on which the rules operate. Jean Krivine developed a stochastic semantics for Biographical Reactive Systems [KMT08b] and suggested possible applications to the modeling of biological systems involving membrane interactions. This work led Jean Krivine to define later an algebraic graph rewriting formalism capturing compartments in the style of bigraphs and adjusted to the specific needs of biological systems [DHK12].

### 4.3 Models of service-oriented programming

A method for writing specifications for distributed software is to describe the behavior by means of contracts. A particular kind of contracts are those defined by using process algebras. The research on contracts is, thus, at the frontier between concurrency theory and service-oriented programming. On the one hand, Castagna, R. De Nicola, and Varacca have studied some aspects of concurrency theory, in particular how to define subtyping relations in the context of higher-order process algebras, resulting in the definition of $C\pi$, an extension of the $\pi$-calculus by semantic subtyping [CDNV08]. On the other hand, Castagna, N. Gesbert, and L. Padovani developed a theory of contracts, that aims at defining in a formal way a notion of “compatibility” between clients and services, as well as ensuring safety aspects of service evolution and replacement [CGP08, CGP09]. Once the theoretical foundations have been settled, the research has followed several directions. One direction aimed at enriching contracts so as they could better model some aspects of service interactions, such as the specification of order preserving asynchronous communications [CP09a]. A second direction aimed at using the theory of contracts to model financial contracts, a work that PPS has developed in collaboration with Lexifi, in the context of Froissart’s master thesis. A last
direction aimed for a better comparison between contracts and session types. The latter provide
an alternative formalism to specify the behavior of services. Castagna, Giachino, M. Dezani, and
L. Padovani, in the context of Giachino’s PhD [Gia09], have developed a semantic characterization
of the theory of session types which, up to that point, had only syntactic definitions [CDCGP09].
This study not only provides a mathematical foundation to session types, but also closes the gap
between them and contracts. Castagna and L. Padovani have extended the theory of contracts with
the main characteristic that contracts lacked with respect to session types, that is, higher-order
capabilities [CP09b].

This has paved the way to the definition of a paradigm for describing global interactions that
could be compatible both with contracts and with session types. This has resulted in the definition
by Castagna, Dezani, and Padovani of global types for multi-party sessions [CDCP12, CDCP11].

4.4 Models for reversibility and causality

► Causal Models. Varacca provided an event structure semantics of the π-calculus. Together
with N. Yoshida he had previously devised an event structure model for a linearly typed version of
the calculus [VY10]. In two steps, Varacca, S. Crafa and N. Yoshida designed a model for the full
calculus. First, a model for the internal fragment of the calculus was proposed [CVY07] which was
later extended to the full calculus [CVY12].

► Reversibility and concurrency. In systems where distributed consensus needs to be found, no so-

*olution can be implemented without letting participants backtrack their local choices. Programming
such consensus in a language that has a native reversible semantics eases both the implementation
and the verification of produced code47. As a first result, Jean Krivine has proposed a benchmark
that compares the “traditional” approach, which consists in implementing the dining philosopher
problem using ad hoc reversible moves, with the one that benefits from the native reversibility of
RCCS, a symmetrization of the operational semantics of Milner’s CCS. The benchmarks show that
the size of the LTS of the ad hoc code is exponential with respect to the size of the specification
LTS, while the RCCS induced LTS scales linearly with the specification [Kri12a].

4.5 Probabilistic systems

The work of Abbes explores probabilistic models for concurrency based on trajectories where events
are partially ordered by time48. The two models he has considered are safe Petri nets and their
associated event structures and trace semigroups.

► Topological and measurable structures for event structures and trace semigroups. The probabil-

*istic layers to be put on top of these two families of models are based on a natural topological
structure. Both the topological and the measurable nature of the space of configurations can be
investigated through a projective limit of finite sets, as shown in [Abb07]. The model of trace sem-
groups consists of the quotient of finitely generated semigroups under trace equivalence relations
of the form \(ab = ba\). The completion of the elements of the semigroup under the order structure
has been the topic of [Abb08].

47In this context of research, Jean Krivine answered the ANR call Ingénierie Numérique et Sécurité 2011 and
obtained the funding of the REVER ANR-11-INSE-007 project of which he is the main coordinator for the PPS
partner.
48This work has been supported by the ANR Blanc project Panda.
Probabilistic models for distributed systems. Abbes elaborated a model for probabilistic Petri nets based on the notion of branching cells. A restriction of his construction was that the corresponding event structure should be locally finite. The generalisation to the general case has been the topic of [AB09]. The ergodic theory of probabilistic Petri nets has been studied in [AB08]. The main contribution of the paper is a Law of Large Numbers (LLN) in the context of asynchronous systems, in a partially ordered time. It is based on the LLN for Markov chains.

Introduction of a new model. A new direction of research has been explored by considering the problem of composing Markov chains. The trace semigroup model is adapted to this problem. Composition of Markov chains has been the subject of [Abb10], currently submitted for publication. The extension of the same problem to more than two chains introduces new technical challenges. The very existence of Markov multi-components processes on more than two sites is non trivial. This first problem has been solved in the research report [Abb11], justifying additional work to come.

4.6 Blazons

A surprising observation made by Manoury is that the blazons used by heraldists to describe emblems can be considered as programs describing very accurately these pictures in a hierarchical way [Man10].

4.7 Package-based software deployment and installation

Software packages are today ubiquitous in software deployment, most prominently in the form of packages known from Free and Open Source (FOSS) software distributions. Managing large amounts of software packages is a major challenge for the editors of software distributions and for the system administrators.

The activities of PPS on the management of large sets of software packages were structured by several cooperation projects. PPS participated in the European (FP 6) research project Environment for the Development and Distribution of Open Source Software (EDOS) which focused on several aspects of software packages from the point of view of the distribution editor and which terminated in July 2007. Di Cosmo from PPS was leader of WP2 which investigated the logical relations between software packages expressed in their so-called meta-data, like dependencies and conflicts. The final phase of the project was dedicated to turn the tools developed by the team into a service integrated into the Debian distribution development.

The EDOS project opened up research questions that have been the basis of the Mancoosi project (Managing the Complexity of the Open Source Infrastructure), which was funded by the European Commission in the FP 7 framework, and ran from February 2008 to May 2011. This project involved six academic partners and four industrial partners from six different countries, and was coordinated by Di Cosmo.

The Mancoosi project focused on managing upgrade problems from the point of view of a system administrator [DCZT08]. PPS contributed in particular to two aspects of the projects. Di Cosmo and Zacchiroli cooperated with the project site L’Aquila on using model-driven techniques to model the state of a system and the effect of executing maintainer scripts, i.e. small customized programs that get executed during installation and removal of software packages [CDRP+09, DRPPZ09].

50 See edos.debian.net
Abate, Di Cosmo, Treinen and Zacchiroli contributed to WP5, of which Treinen from PPS was the leader, on the organization of an international competition of upgrade problem solvers. They cooperated with the industrial partners of the project in the establishment of a data base of upgrade problem instances [AT11a], and organized an international competition of solvers [AT11b, ACTZ12] which took place in 2010, 2011 and 2012. An important outcome of this project was the definition of CUDF as a common data interchange format for upgrade and installation problems [ADCTZ11, ACTZ12]. CUDF allows to encode several existing formats of package metadata like the ones used by Debian, RedHat and Eclipse [AGL+10], and also problems from related areas like feature diagrams in software product lines [DCZ10]. Furthermore, a prototype of a domain-specific language for defining user preferences in solving upgrade problems was proposed [TZ09].

An important line of work consists in analyzing complex package relationships according to the semantic properties in the context of a complete package repository. This program was started by Abate, Boender, Di Cosmo and Zacchiroli with the works on so-called strong — i.e. logically implied — conflicts and dependencies [ABDCZ09, CB10, Boe11]. Di Cosmo and Vouillon have shown how semantic properties of package relations can be exploited in order to make the logical problems of package installability representable in a very compact and understandable way. The algorithm behind the coinst tool [VDC11] yields a very understandable representation of a package repository, Abate, Di Cosmo, Treinen and Zacchiroli have shown how to detect logical flaws in package repositories by using temporal properties [ADC11, ADCTZ12].

Production and dissemination of software is an integral part of research that aims at improving software tools. The research performed in this theme has lead to the creation of software tools that are all available as Free Software, like the coinst tool developed by Vouillon, the libcudf library developed by Zacchiroli, and the dose library and toolsuite developed mainly by Abate and Boender which comprises a collection of software that results form the EDOS and Mancoosi projects. Dissemination activities target both the scientific community, as with the series of LoCoCo workshops (Logics for Component Configuration) initiated by the Mancoosi project, and the FOSS community through journals [Tre09] and meetings [TZ08, DCDB+08] of the FOSS community.

The line of work started with EDOS and Mancoosi finds its natural continuation in two ongoing collaborative projects, Dorm and Aeolus.

4.8 Programming for the Web

The main goal of this research, performed by Balat and his group, is to adapt Web programming techniques to the current evolution of the Web where browsers have become powerful virtual machines in which more and more programs are executed. The approach consists mainly in using abstractions to provide higher level programming techniques for the Web, and employing sophisticated typing features to create reliable Web applications and make them easy to maintain. Finally

51http://www.mancoosi.org/misc/
52coinst.irill.org
53http://www.mancoosi.org/cudf/
54http://www.mancoosi.org/software/#index2h1
55The DORM project is applying the expertise on components dependencies to industrial problems related to the management of software repositories, in particular in the Java world.
56The Aeolus project, funded by ANR in the Arpege program, aims at lifting the results obtained to the even more complex situation of computing services provided by a dynamic collection of machines.
they provide a new unified approach to program both the client and server parts of a Web application as a single program.

Most of the research results of the group are implemented and distributed widely as free software in the Ocsigen Web framework for the Ocaml language [BVY09]. Ocsigen is now used by several companies, like BeSport or MetaWeb (both San Francisco), Li-Cor (Lincoln, Nebraska), Hypios (Paris), etc. Several startups are currently being created using the framework.

**Programming traditional Web interaction.** The first of their results concerns traditional Web interaction, that is: programming links, forms, back button, bookmarkable pages, etc. This resulted in proposing a new model called service based Web programming. A service is a kind of function invoked in reaction to an HTTP request and that usually returns a Web page. The most striking feature is the service identification mechanism they propose [Bal09a, Bal09b], that is, the way the server choses the service to be executed to handle a request. Another very convenient feature is the ability to dynamically create new services, for example services customized to one user and that depend on previous interactions with him. This simplifies a lot programming sequences of pages, each depending on the previous ones and is usually known as continuation based Web programming. This service model has been implemented for Ocaml in the Eliom module of Ocsigen. They also worked on filling the gap between Eliom and the CDuce language by working on a CDuce version of Eliom, called OcCDuce.

**Executing programs in a Web browser.** Today, most Web developers are using distinct languages for programming traditional Web interaction (e.g. forms, links and bookmarks) and client side features.

A significant step towards the implementation of Web applications as a single Ocaml program is to allow for Ocaml programs to run inside a browser. Two techniques have been experimented: one consists in programming in Javascript an Ocaml virtual machine, to run Ocaml bytecode programs inside a browser without recompiling them [CBC08, CCV12]. The second one consists in writing a compiler from Ocaml to Javascript (called Ocsigen Js of Ocaml) More precisely, an original approach has been followed, consisting in taking Ocaml’s bytecode as the source of the compiler [CCV12] which becomes very easy to maintain (as the bytecode is not evolving a lot) and also very easy to use: any Ocaml program may now be compiled to Javascript, to be used in a browser. This compiler produces programs with excellent performance on modern browsers, they run typically faster than the original bytecode version.

**Client-server Web applications in Ocaml.** Balat, Chambart and Henry developed the possibility of writing client-server Web applications as a single program [BCH12]. Web programs are written like ordinary desktop applications, interfaces are built in the usual declarative way and sent together with the code corresponding to dynamic parts.

**Typed Web programming.** Ocsigen’s developers faced a few missing features in Ocaml’s serialisation mechanism. Chambart and Henry proposed two patches on the Ocaml compiler: one to allow serialising closures with dynamic linking, and the other one introducing some kind of dynamic types. An essential use of static typing in Ocsigen is for checking at compile time the validity of functions generating HTML, thus guaranteeing that Ocsigen programs always generate pages that respect the W3C recommendations [Can11].

---

57 ANR project PFW.
58 ANR project Codex, Section 2.2.
59 This research was supported by the DIM logiciels et systèmes complexes project PWT.
60 This research is supported by the ANR project PWD.
Other software projects. Besides Eliom and Js of OCaml, the Ocsigen team wrote several other pieces of software: Lwt, a monadic cooperative threading library which is now widely used amongst Ocaml’s community (Vouillon)
61; Ocsigen server, an extensible Web server written in Ocaml (Balat, Vouillon, Yakobowski); Macaque, a library for type safe database queries in Ocaml using comprehensions (Vouillon, Scherer); O’Closure, a binding for the Google Closure Library (Cardoso, Vouillon); Ocsimore, a wiki and content management system (Balat, Yakobowski, Becker)
62.

4.9 Networks, Concurrency

IPv6. Since 1996, Legrandgérard is a member of the G6
63 group, whose objective is to experiment and implement the new internet protocol IPv6 which will replace the current IPv4 protocol whose stock of addresses is almost exhausted. He was very active in the IPv6 Renater 2 project which led to the creation of a native IPv6 network within the Renate
64 infrastructure. He has worked in particular on the DNSv6 service (Domain Name Service with IPv6 extensions).

Continuation Passing C. CPC
[KC11, KC12a, Chr12] is an extension of the C programming language designed to easily write extremely lightweight and efficient concurrent programs, most notably network servers and peers in distributed systems. CPC is implemented as a series of source-to-source transformations familiar from programming language semantics, notably λ-lifting and a CPS transform. Although CPC is an extension of C, a language notoriously hostile to formal treatment, the correctness of the core of CPC has been formally proved
[KC12a]. The most difficult part of this proof is the correctness of (a particular case of) λ-lifting in a CBV imperative language, a novel result of intrinsic theoretical interest
[KC12b].

In order to show that CPC is usable in practice by non-specialists, Chroboczek and Kerneis have supervised two undergraduate students writing a highly-concurrent implementation of the BitTorrent file transfer protocol in it. The resulting program, called Hekate, is of production quality, and has successfully (and legally) served gigabytes of data over many weeks.

BitTorrent DHT. One of the many sub-protocols found in BitTorrent uses a distributed hashtable (DHT) using a variant of the Kademlia algorithm. Hekate uses a home-brew implementation of the DHT sub-protocol (written by Chroboczek), which has been separated into a stand-alone library. This work has lead to the publication of the specification of the BitTorrent DHT’s operation over IPv6
[Chr09], a specification implemented by multiple implementations of BitTorrent.

Babel and Babel-Z. Chroboczek’s experiments with large-scale distributed systems have lead to the design of Babel, a distance-vector routing protocol designed to be efficient and robust in both classical wired networks and wireless mesh networks (MANETS)
65. Babel has gone through a number of iterations, but the basis has been stable for over two years now; the protocol has been published as an IETF
66 “Request for Comments” (RFC), RFC 6126. In small networks, Babel has been shown to be more efficient than the best community-driven protocols for mesh networks. While Chroboczek is aware of a number of evaluations of Babel in larger networks (up to a few

61 This research was supported by the ANR Jeunes chercheurs, jeunes chercheuses project Programmation fonctionnelle pour le web.
62 The full list of software projects of this group is available on http://ocsigen.org.
63 See http://g6.asso.fr/.
64 National Network for Education and Research.
65 This research has been supported by the PICRI programme of Région Île de France.
66 http://www.ietf.org/
thousand nodes), these are proprietary industrial deployments and the results have not been made available to him.

Babel-Z is an experimental variant of Babel which is able to use multiple radio frequencies in order to load-balance traffic in wireless mesh networks. Routing on varying frequencies raises a number of difficult theoretical problems (the *isotonicity* of the routing metric is violated) which have been little studied and are not well understood. Initial experiments with Babel-Z show promising results.

### 4.10 Software safety

Recently, Esterel Tech. decided to use the Ocaml language for the development of a code generator for critical embedded systems (typically for avionics and other real time critical applications). Manoury acted as an expert in functional programming languages, writing a technical report on code traceability in the Ocaml compiler. This report has served as a basis for Esterel Tech.’s engineers to produce the test sets and documentations required by the civil avionics authorities [PAM+09]. A natural approach to produce secure programs for real time critical embedded applications consists in using previously certified components. Manoury worked on the extension the refinement model of method B in order to take such components into account\(^{67}\).

### 5 Appendix: Publications and Software productions

Names of co-authors not affiliated to PPS are written in italic.

**Journals**


---

\(^{67}\)This research is supported by the ANR Arpège Cercles 2 project.


<table>
<thead>
<tr>
<th>Reference</th>
<th>Title</th>
<th>Journal/Conference</th>
<th>Year</th>
<th>DOI</th>
</tr>
</thead>
</table>


<table>
<thead>
<tr>
<th>Reference</th>
<th>Title and Authors</th>
</tr>
</thead>
</table>


**Invited talks in international conferences with proceedings**


Other invited talks


Peer-reviewed international conferences with proceedings


Pierre Clairambault. Estimation of the length of interactions in arena game semantics. In Foundations of Software Science and Computational Structures - 14th International Conference, FOSSACS 2011, Held as Part of the


Peer-reviewed national conferences with proceedings


Communications at national or international conferences without proceedings


**Editing of collective works**


**Book chapters**


Scientific popularization works


PhD Thesis


**Technical reports**


premier numéro de la revue De scientia (revue semestrielle du groupe de recherche franco-italien "Pensée des sciences" (dir. Charles Alunni, ENS Paris, et Mario Castel- lana, Università degli studi di Bari), qui était censé paraître en 2008. Patient, l’auteur du présent article en est encore à se demander si le premier numéro de cette revue paraîtra un jour.


Other publications


[HS] Hugo Herbelin and Alexis Saurin. $\lambda\mu$-calculus and $\Lambda\mu$-calculus, a capital difference. submitted to APAL.

Publications prior to the affiliation to PPS


Software development

[AARGa] Roberto Amadio, Nicolas Ayache, and Yann Régis Gianas. (Prototype) Within the CerCo project (Section 6.4), Amadio has developed with Ayache and Regis-Gianas tools to certify and reason on the execution cost of C programs and functional programs.

[AARGb] Nicolas Ayache, Roberto Amadio, and Yann Régis-Gianas. Cerco is a cost annotating compiler for C.

[ABDC+] Pietro Abate, Jaap Boender, Roberto Di Cosmo, Ralf Treinen, and Stefano Zacchiroli. Dose, a software library and a tool suite for the analysis of software package repositories. This software is published under the LGPL.

[ACCK] Pejman Attar, Yoann Canal, Juliusz Chroboczek, and Gabriel Kerneis. Hekate is a highly-concurrent BitTorrent seeder.

[ARG] Christine Audebaud, Philippe Paulin-Mohring and Yann Régis Gianas. (Prototype) In 2008, in collaboration with P. Audebaud (team Plume of LIP lab, ENS Lyon) and C. Paulin-Mohring (LRI lab of Université Paris Sud and INRIA team Proval), Régis-Gianas worked on proofs of probabilistic programs. He extended the Why proof system with randomized primitives in the programming language and predicates over random distributions in the specification language in an experimental version of this system.

[Bal] Vincent Balat. Ocsforge is a collaborative software development platform (forge). Based on Ocsigen and Ocsimore.

[BBM] David Baelde, Romain Beauxis, and Samuel Mimram. Liquidsoap is a scripting language for generating audio Streams.

Jaap Boender and Berke Durak. **Tart** is a tool to repartition an open-source software distribution over several media (CDs, DVDs).

Jaap Boender. **Ceve** is a generalized open-source software package parser and translator.

Vincent Balat and Jérôme Vouillon. **Eliom** is a framework for programming client/server Web applications in **Ocaml**.

Vincent Balat, Jérôme Vouillon, Boris Yakobowski, Stéphane Glondu, and Gabriel Kerneis. **Ocsigen** server is a Web server developed in **Ocaml**, with the LWT cooperative thread library. It has been designed in order too been extended easily by writing **Ocaml** modules. Many extensions already exist (like a module for executing CGI script, a reverse proxy, access control, redirections, URL rewriting, page compression, user configuration files, etc.).

Benjamin Canou. **O’Browser** is an **Ocaml** virtual machine written in **Javascript**, to run **Ocaml** program in Web browsers.

Roberto Di Cosmo, Vincent Balat, and Jean-Vincent Loddo. **Demolinux** is one of the first **Linux** Live-CD.

Emmanuel Chailloux. **OCamIL** is an experimental **Ocaml** compiler that targets Microsoft **.NET**.

Emmanuel Chailloux. **O’Jacare** is a code generator to help in interoperability between **Java** and **Ocaml** through their object model.

Juliusz Chroboczek. **AHCP** is an autoconfiguration protocol for **IPv6** and dual-stack **IPv6/IPv4** networks designed to be used in place of router discovery and **DHCP** on networks where it is difficult or impossible to configure a server within every link-layer broadcast domain, for example mobile ad-hoc networks.

Juliusz Chroboczek. **Juppix** is a LiveCD **GNU/Linux** based on **Knoppix** including programs used by **Université Paris Diderot L** students.

Juliusz Chroboczek. **Polipo** is a caching web proxy.

Juliusz Chroboczek. **Stochastic Fair Blue (SFB)** is an active queue management algorithm for packet routers that attempts to simultaneously bound queue length (and hence latency), minimise packet drops, maximise link utilisation, be fair to reactive aggregates, reliably detect non-reactive aggregates (aggregates that don’t respond to congestion control indications) and put them into a so-called penalty box.

Juliusz Chroboczek and Gabriel Kerneis. **Babel** is a distance-vector routing protocol for **IPv6** and **IPv4** with fast convergence properties. It is based on the ideas in **DSDV**, **AODV** and Cisco’s **EIGRP**, but uses a variant of **ETX** link cost estimation rather than a simple hop-count metric.

Juliusz Chroboczek and Gabriel Kerneis. **CPC** is a programming language designed for writing concurrent systems. The **CPC** programmer manipulates cooperatively scheduled
threads; the CPC program is then processed by the CPC translator, which produces highly efficient event-loop code. In the author’s opinion, this approach gives the best of the two worlds: the convenience of programming with threads, and the low memory usage of event-loop code.

[Cos] Roberto Di Cosmo. OCamlP3l is a compiler for Caml parallel programs.

[FCA] Alain Frisch, Giuseppe Castagna, and Pietro Abate. CDuce is a modern XML-oriented functional language with innovative features. A compiler is available under the terms of an open-source license. CDuce is type-safe, efficient, and offers powerful constructions to work with XML documents.

[Gim] Stephane Gimenez. Laby is a teaching software: learn how to program with ants and spider webs.


[HV] Haruo Hosoya and Jérôme Vouillon. XDuce is a typed programming language that is specifically designed for processing XML data.

[Kri] Jean Krivine. KaSim is a stochastic simulator for the $\kappa$-calculus, a rule-based language for the modeling of protein-protein interaction networks.

[PVJ] Benjamin Pierce, Jérôme Vouillon, and Trevor Jim. Unison is a file synchroniser.

[RGa] Yann Régis-Gianas. Menhir is an LR parser generator for Ocaml.


[VCR] Julien Verlaguet, Emmanuel Chailloux, and Vivien Ravet. HirondML is an experimental Ocaml library that implements migrating threads. To minimize data transfers when a thread migrates, HirondML distinguishes local variables from global ones: the former are copied during the migration process, whereas the latter are rebound in the target process.

[VD] Jérôme Vouillon and Jérémie Dimino. Lwt is a cooperative threads library for Ocaml, written in monadic style.

[Voua] Jérôme Vouillon. Debcheck is a tool to check for broken packages in an open-source software distribution (e.g. Linux, FreeBSD).


[VTCB] Jérôme Vouillon, Ralf Treinen, Roberto Di Cosmo, and Jaap Boender. EDOSLib is a library which provides the foundation of some tools that are used in the framework. In particular EDOSLib implements: an object model for representing package repositories
information and their structure; a set of functionalities to explore manage the package repository structure (e.g. extracting subrepositories and dependency closures); EGraph input/output functionalities.

[YB] Boris Yakobowski and Vincent Balat. Ocsimore is a content management system written with Eliom, containing a Wiki, forum, blog, news, etc (under development)\(^68\).

6 Research Projects

The PPS lab is and has been involved in many research projects during the period 2007-2012. We provide here a comprehensive list, with references to pages of this activity report where the corresponding research is described.

6.1 Université Paris Diderot projects

PPS lab and IMJ initiated two interdisciplinary partnerships:

- *Sémantique, algèbre opéradique et catégorication: un point de convergence novateur entre informatique, mathématique et physique* in 2010.

Both projects were supervised and animated by Melliès and received a funding of 15 000€ from Université Paris Diderot, used for invitation and organization of small meetings. These projects have allowed to reinforce the cooperation between the two laboratories on the above mentioned topics.

6.2 CNRS groupements de recherche

Several members of the PPS lab take part to the activities of the CNRS GDR-IM\(^69\), and more specifically to the working groups GEOCAL and LAC, this latter being animated and organized by Kesner. The PPS lab participates also to the GDR Topologie Algébrique et Applications\(^70\), as a team led by Gaucher. In that way, the PPS lab deepens its rich connections with the Mathematics community on topics such as higher dimensional algebra and homotopy. Last, the PPS lab is a team of the GDR Théorie des jeux: modélisation mathématique et applications; Melliès is particularly active in this GDR.

6.3 National projects

  \footnote{Ocsimore is used on \url{www.pps.univ-paris-diderot.fr}.}  
  \footnote{Informatique Mathématique \url{http://www.gdr-im.fr/}, directed by Brigitte Vallée.}  
  \footnote{See \url{http://math.univ-lille1.fr/~gdrtop05/index.html}.}  
  \footnote{ANR: Agence Nationale pour la Recherche, \url{http://www.agence-nationale-recherche.fr/}}
• INRIA ARC project CeProMi. Partners: team ProVal of INRIA Saclay, team DCS of the Verimag lab (Université Joseph Fourier), team Gallium of INRIA Rocquencourt, team Cassis of INRIA Besançon and Nancy. Local leader: Yann-Régis Gianas. Started in 2008, ended in 2009. Local funding: 30 000€.
http://www.lri.fr/cepromi/

• ANR Arpège project Cercles 2 (Certification compositionnelle des logiciels embarqués et sûrs). Partners: team APR of the LIP6 lab (Université Pierre et Marie Curie), companies Sagem Défense et Sécurité, Clearsys. Local leader: Manoury. Number of local researchers (men months): 12. Local funding: 139 123€. Started in 2011, ending in 2014 (44 months). Quality label of competitiveness pole SYSTEMATIC Paris Région. See page 43.
http://www.algo-prog.info/cercles/

• ANR Blanc project Choco (Curry-Howard pour la concurrence). Partners: team LCR of the LIPN lab (Université Paris Nord – Villetaneuse), team LDP of the IML (Université d’Aix-Marseille 2), LIF lab (Université d’Aix-Marseille 1), team LIMD of Lama lab (Université de Chambéry), team Plume of the LIP lab (ENS Lyon). Leader: Ehrhard (PPS). Number of local researchers (men months): 24. Local funding: 134 974€. Started in 2007, ended in 2011 (42 months). See page 24.
http://choco.pps.jussieu.fr/

• ANR Domaines émergents Codex (Efficiency, Dynamicity and Composition for XML: Models, Algorithms and Systems). Partners: INRIA Grenoble (team WAM, and LIF lab of Université d’Aix-Marseille 1), team Mostrare of INRIA Lille, team Gemo of INRIA Saclay, company Innovimax SARL, Laboratoire d’informatique), Université Paris Sud (BD of the LRI lab (Université François Rabelais). Local leader: Castagna (PPS). Number of local researchers (men months): 20. Local funding: 118 224€. Started in 2009, ended in 2012 (48 months). Quality label of competitiveness pole SYSTEMATIC Paris Région. See page 41.
http://codex.saclay.inria.fr/

http://www-lipn.univ-paris13.fr/~mazza/Collodi/

http://compcert.inria.fr/

• INRIA ADT72 project Coq. Partners: INRIA teams TypiCal, ProVal, Marelle and pi.r2. Leader: Herbelin (PPS). Started in 2009. It supports visits and meetings between developers and

\footnote{\textit{Action de Développement Technologique}}
aims at strengthening the community of Coq users and contributors. Four national-level Coq meetings have been organized as part of this ADT (two in 2009 and two in 2010). The ADT has also supported two Coq Asian summer schools organized by Jean-Pierre Jouannaud in Beijing (one in 2009 and one in 2010). A new ADT about Coq has been launched in 2011.

http://coq.inria.fr/adt/


http://eternal.cs.unibo.it/

- FEDER domaine d’intérêt majeur (DIM) Région Île de France Logiciels et Systèmes Complexes (LSC) project DORM. Partners: companies NUXEO, ZENIKA. Local leader: Di Cosmo. Number of local researchers (men months): 24. Local funding: 120 120€. Started in 2010, ending in 2012 (24 months). Quality label of competitiveness pole SYSTEMATIC Paris Région. See page 40.

http://www.systematic-paris-region.org/fr/projets/dorm-0

- ANR Investissements d’avenir recherche, bio-informatique project Iceberg (Des modèles de population aux populations de modèles: observation, modélisation et contrôle de l’expression génique au niveau de la cellule unique). Partners: team Contraintes of INRIA Rocquencourt, Institut Jacques Monod and MSC lab (Université Paris Diderot), team BM2A of the CGPHMC lab (Université Lyon 1), LIFL (Université de Lille 1). Local leader: Jean Krivine. Number of local researchers (men months): 24. Local funding: 40 976€. Started in 2011, ending in 2016 (60 months). See page 35.

http://www.lix.polytechnique.fr/~lutz/orgs/infer.html


http://www.lix.polytechnique.fr/~lutz/orgs/infer.html

- ANR Blanc project Inval (Invariants algébriques des systèmes informatiques). Partners: team MeaSI of LIST lab of CEA, team LDP of the IML lab (Université d’Aix-Marseille 2), team GTA of the I3M lab (Université de Montpellier), team Algèbre, topologie, groupes quantiques, représentations of the IRMA lab (Université de Strasbourg), team Calligramme of the LORIA lab (Université de Lorraine), Algèbre, géométrie, logique team of the ICJ lab (Université Lyon 1). Local leader: Gaucher. Number of local researchers (men months): 48. Local funding: 30 000€. Started in 2005, ended in 2008. See page 34.

http://www.pps.univ-paris-diderot.fr/~inval/

- ANR Blanc project Logoi (Logique et géométrie de l’interaction). Partners: team LDP of the IML lab (Université d’Aix-Marseille 2), team LCR of the LIPN lab (Université Paris Nord


- ANR Blanc project Panda (Parallel and Distributed Analysis). Partners: teams Comete and Parsifal of INRIA Saclay, LIST lab of CEA, team LCR of the LIPN lab (Université Paris Nord – Villetaneuse), team Infini of the LSV lab (ENS Cachan), team Plume of the LIP lab (ENS Lyon), team LIMD of the LAMA lab (Université de Chambéry), team Comète of the LIX lab (École Polytechnique), team Move of the LIF lab (Université d’Aix-Marseille 1), team LDP of the IML lab (Université d’Aix-Marseille 2), company Airbus. Local members: Abbes and Melliès. Number of local researchers (men months): 27. Local funding: 23 000€. Started in 2008, ended in 2011. See page 38.  
  http://www.lipn.univ-paris13.fr/~mazza/Panda/

  http://www.di.ens.fr/~zappa/projects/parsec/

  http://paral-itp.lri.fr/


  http://www-sop.inria.fr/indes/pwd/
6.4 European projects and partnerships

  
  http://iml.univ-mrs.fr/~vaux/quand/

- **ANR Blanc** project Récré (Réalisabilité pour la logique classique, la concurrence, les références et la réécriture). Partners: team Plume of the LIP lab (ENS Lyon), team LDP of the IML lab (Université d’Aix-Marseille 2), team LIMD of the LAMA lab (Université de Chambéry). Local leader: Herbelin. Number of local researchers (men months): 28. Local funding: 226 720€. Started in 2011, ending in 2015 (48 months). See page 32.
  
  http://recre.ens-lyon.fr/

  
  http://www.pps.univ-paris-diderot.fr/~jkrivine/REVER/

- **ACI Masses de données** project TraLaLa (XML Transformation Languages: logic and applications). Partners: teams Gemo and Mostrare of INRIA Futurs, team Vérification of LIAFA lab (Université Paris Diderot), team Langages of LIENS lab (ENS Ulm), team Move of LIF lab (Université d’Aix-Marseille 1), team STC of LIFL lab (Université de Lille 1), team Bases de Données of LRI lab (Université Paris Sud). Local leader: Castagna. Started in 2007, ended in 2010 (48 months)\(^\text{73}\). See page 18.
  
  http://www.cduce.org/tralala.html

  

\(^\text{73}\)This project was initiated by G. Castagna when he was still a member of the LIENS lab, and has been administratively managed from LIENS until its ending.
• PHC \textsuperscript{74} PAI \textsuperscript{75} Germaine de Staël (Suisse) project Deep Inference and the Essence of Proofs. Partners: team Parsifal of INRIA Saclay, Group for theoretical computer science and logic of University of Bern. Local leader: Parigot. Local funding: 3 250€. Started in 2006, ended in 2007. See page 27.


• CNRS-CNR\textsuperscript{76} Coopération bilatérale avec l'Italie project Interaction et Complexité 2. Other partner: Department of Computer Science of University of Bologna. Local leader: Olivier Laurent. Started in 2006, ended in 2007.


• PHC \textit{Alliance} (UK) project Nouveaux systèmes de déduction et méthodes de normalisation pour l’inférence profonde. Other partner: Department of Computer Science of Bath University. Local leader: Parigot. Started in 2006, ended in 2007. See page 27.

• PHC \textit{Amadeus} (Austria) project Outils logiques d’analyse des programmes. Other partner: Institute of Computer Science (University of Innsbruck). Local leader: Parigot. Started in 2009, ended in 2010. See page 27.


\textsuperscript{74} Partenariat Hubert Curien
\textsuperscript{75} Programme d’Action Intégrée
\textsuperscript{76} CNR: Consiglio Nazionale della Ricerca (Italy)
• MIUR\textsuperscript{77} Internazionalizzazione del sistema universitario project Réseau italo-français de recherche en logique et géométrie du calcul. Partners: Department of Philosophy of University of Roma 3, Department of Computer Science of University of Bologna, Department of Computer Science of University of Torino, Department of Computer Science of University of Roma La Sapienza, Department of Pure and Applied Mathematics of University of Padova, team LDP of the IML (Université d’Aix-Marseille 2), team Calligramme of LORIA (Université de Lorraine), team LCR of LIPN (Université Paris Nord – Villetaneuse). Coordinator: University of Roma 3. Local leader: Bucciarelli. Started in 2006, ended in 2008. Local funding: 10 000€. See page 25.


6.5 Extra-european projects and partnerships


• International Laboratory INFINIS. Partners: Université Paris Diderot, CNRS, CONICET. Leaders: Kesner (France) and Sergio Yovine (Argentina). Started in 2011. Local funding: 30 000€. See page 18.


• INRIA associate team Semacode (Stratégies d’évaluation, machines abstraites et contrôle délimité). Partners: team of Zena Ariola of University of Oregon, team Focus (located in Bologna, Italy) of INRIA Sophia Antipolis, team of Silvia Ghilezan of University of Novi Sad. Leader: Saurin.

\textsuperscript{77}Italian ministry of universities and research


- **Programme franco-chinois de recherche avancée** project *Types, processus et leurs applications en sécurité et systèmes biologiques*. Partners: Team DCS of Verimag lab (Université Joseph Fourier), Institute of Software Chinese Academy of Sciences, Department of Computer Science of Shanghai Jiao Tong University. Local leader: Curien. Started in 2006, ended in 2008. See page 35.


### 6.6 Long term collaborations

We provide here a list of long term cooperations involving members of the PPS lab. Some of these collaborations take place in projects listed above.

- Balat is collaborating with Emmanuel Chailloux (LIP6) on several projects: ANR PWD, *domaine d’intérêt majeur (DIM) Région Île de France Logiciels et Systèmes Complexes (LSC)*, PhD advising, workshop organisation.

- Berline collaborates regularly with Antonino Salibra (Ca’Foscari University, Venice) and Klaus Grue (University of Copenhagen).

- Bucciarelli has a regular collaboration with Antonino Salibra (Ca’Foscari University) and with Simona Ronchi della Rocca (University of Torino).

- Castagna collaborates regularly with Imperial College London (Nobuko Yoshida in particular), University of Torino (Mariangiola Dezani and her group), Université Paris Sud, INRIA Grenoble.

- Curien has a continuing collaboration with Glynn Winskel and Marcelo Fiore (Cambridge University), mainly on game semantics and categorical semantics.

- Di Cosmo has been regularly collaborating with various partners from different laboratories, and has published with colleagues from University of Bologna, University of Pisa, University of Torino and Université de Nice Sophia Antipolis.

---

78 A research program of the French ministry of foreign affairs.
• Ehrhard has a continuing collaboration with Lorenzo Tortora de Falco (Department of Philosophy, University of Roma 3), Michele Pagani (LIPN, Université Paris Nord – Villetaneuse), Thomas Streicher (Department of Mathematics, Technische Universität Darmstadt) and Antonino Salibra (Department of Computer Science, Ca’Foscari University). He has also regular scientific interactions with Martin Hyland (Department of Pure and Applied Mathematics, Cambridge University).

• Faggian has regular collaborations, leading to a number of visits and exchanges, with team Focus of INRIA Sophia-Antipolis and University of Bologna (Ugo dal Lago), and with RIMS lab of University of Kyoto (Michele Basaldella and Kazushige Terui).

• Harmer and Jean Krivine are engaged in an active collaboration with Walter Fontana of the Department of Systems Biology at Harvard Medical School on the application of rule-based modeling to cellular signaling networks. They are also engaged in a collaboration with Glynn Winskel (Cambridge University) and Danos (University of Edinburgh) on the graph-theoretic and categorical foundations of the rule-based approach.

• Herbelin has regular collaborations with the University of Oregon (Zena Ariola), University of Novi Sad in Serbia (Silvia Ghilezan), University of Anseong in Korea (Gyesik Lee).

• Kesner has regular collaborations with University of Torino (Simona Ronchi della Rocca), National University of Quilmes (Eduardo Bonelli) and University of Buenos Aires (Alejandro Ríos) in Argentina, University of Sydney (Barry Jay), LIX lab of École Polytechnique (Beniamino Accattoli and Stéphane Lengrand).

• Apart from the above mentioned collaboration with Harvard Medical School, University of Edinburgh and Cambridge University, Jean Krivine has a constant collaboration with LIENS lab of INRIA-ENS Ulm (Jérôme Férat), mainly on the development of KaSim.

• Melliès has a regular collaboration with Martin Hyland (Cambridge University) on game semantics and higher dimensional algebra, with Hasegawa Masahito (RIMS, Kyoto, Japan) on categorical semantics of programming languages, and with Andrew Appel (Princeton), Nick Benton (Microsoft Research) and Zhong Shao (Yale) on the semantics of low level languages.

• Métayer collaborates regularly with Dimitri Ara (IMJ, Université Paris Diderot and Université Pierre et Marie Curie), Yves Guiraud\(^79\) and Philippe Malbos (ICJ, Université Lyon 1), Yves Lafont (IML, Université d’Aix-Marseille 2) and Krzysztof Worytkiewicz (LAMA, Université de Chambéry).

• Parigot collaborates regularly with Matthias Baaz (Technische Universität Wien), Alessio Guglielmi (University of Bath), Georg Moser (University of Innsbruck), Revaz Grigolia (Tbilissi State University).

• Régis-Gianas is close to the teams ProVal and Gallium of INRIA.

• Ruet has been affiliated with the IML (Université d’Aix-Marseille 2) until August 2011. He is still collaborating with members of the IML.

\(^{79}\)Member of the PPS lab since January 2012
• Saurin has regular collaborations with members of LIX (École Polytechnique), IML (Université d’Aix-Marseille 2), LIP (ENS Lyon), LIPN (Université Paris Nord – Villetaneuse) in France and with members of RIMS (University of Kyoto), University of Oregon, University of Novi Sad (Serbia), University of Torino.

• Tasson collaborates regularly with Michele Pagani of LIPN (Université Paris Nord – Villetaneuse), Lionel Vaux of IML (Université d’Aix-Marseille 2) and Richard Blute (Department of Mathematics of University of Ottawa).

• Treinen continued, after his move from ENS Cachan to PPS, to collaborate with Florent Jacquemard, Steve Kremer, Étienne Lozes and Jules Villard from the LSV lab in Cachan. He also has a continuing cooperation with Hitoshi Ohsaki, and since more recently with Cyrille Artho, from AIST Kansai in Amagasaki, Japan.

• Varacca has an ongoing continuous collaboration with Silvia Crafa of the Department of Mathematics of University of Padova.

• Zacchiroli collaborates regularly with Yacine Boufkhad of the LIAFA. He collaborates regularly with Gabriele D’Angelo, Angelo Di Iorio, Davide Rossi, Fabio Vitali, and Gianluigi Zavattaro, members of the Department of Computer Science of University of Bologna. He has collaborated regularly with Antonio Cicchetti, Davide Di Ruscio, Patrizio Pelliccione, and Alfonso Pierantonio members of the Department of Computer Science of University of L’Aquila.

7 Organization of national and international conferences and meetings

7.1 Organization of two major international conferences

► RDP’2007, June 25th – June 29th, 2007. In 2007, the PPS lab (Bucciarelli and Padovani), LSV (Treinen, who was not yet a member of PPS) — ENS Cachan — and CÉDRIC (X. Urban) — ENSIIE — organized the Federated Conference on Rewriting, Deduction and Programming (RDP’2007), which included this year RTA (Rewriting Techniques and Applications), TLCA (Typed Lambda-Calculus Applications), a colloquium in honor of Giuseppe Longo and 8 workshops. Each of these events had its own program committee. The conference took place in the buildings of the Conservatoire National des Arts et Métiers in the third district of Paris and gathered more than 280 attendees. The event has been sponsored by CNRS, INRIA, Région Île de France, ENSIIE and Conservatoire National des Arts et Métiers.

► LC’2010, July 24th – July 31st, 2010. In 2010, the PPS lab and the Équipe de Logique Mathématique (Université Paris Diderot and CNRS), together with the LIAFA, organized the Logic Colloquium80, which is the main international conference on Logic in general, with an emphasis on Mathematical Logic and Logic for Computer Science. It is the official yearly conference of the Association for Symbolic Logic (ASL). Curien (PPS) was a member of the program committee, and the organizing committee was co-chaired by Ehrhard and included Abbes, Ainardi, Balat, Legrandgérad, Manoury, Manuset81 and Rozière. The PPS administrative staff (consisting of

81Former member of the administrative staff of the PPS lab.
Ainardi and Mansuet) dealt with most of the administrative aspects of the organization. The colloquium gathered about 300 attendees and had an important grant programme for students and young researchers. It took place at the Halle aux Farines, one of the buildings of Université Paris Diderot. The conference was sponsored by many institutions: CNRS, INRIA, Région Île de France, Ville de Paris, ASL and of course University Paris Diderot.

### 7.2 Other conferences and events


- Curien was president of the Organisation Committee for *Jean-Yves Girard’s Days* (anniversary meeting for his 60th birthday), Institut Henry Poincaré, Paris, September 2007.

- Curien and Melliès organised the workshop **GGJJ2011**\(^{82}\), in honour of Gérard Berry and Jean-Jacques Lévy, in Gérardmer (February 2011).

- Curien has created (together with Jean-Eric Pin of LIAFA) a permanent Committee for the organisation of the annual *Spring School in Theoretical Computer Science*, in 2002, and has been a member of this committee until 2012.

- Faggian has been co-organizer (with K. Terui of RIMS, University of Kyoto) of the meeting *Geometry of Interaction* Kyoto, 7-11 November 2011; she has been co-organizer (with O. Laurent and M. Quatrini) of the week 2 of *Logic and Interaction 2012*\(^{83}\) CIRM, Marseille 6-10 February 2012.

- Harmer co-organized, with D. Ghica (University of Birmingham, UK), the 3rd International Workshop on *Games for Logic and Programming Languages* (GaLoP 2008), a satellite workshop of *ETAPS 2008*\(^{84}\), in Budapest and co-organized, with J. Lynch (Clarkson University, USA), the 3rd LICS Workshop on *Logic for Systems Biology* (LSB), a satellite workshop of *PLoC 2010*, in Edinburgh\(^{85}\).

- Herbelin initiated the *Coq workshop* in 2009. The first venue held as a satellite event of *TPHOLs 2009* in Munich, Germany\(^{86}\). With V. Kuncak and K. Heljanko, he organized the workshop *AIPA* as a satellite event of *ETAPS 2012*, Tallinn, Estonia\(^{87}\). Herbelin, Saurin and Zeilberger organized the *TPDC* workshop in 2011 as a satellite event of *RDP 2011* in Novi Sad, Serbia\(^{88}\).

- Kesner organized two workshops of the working group *Logique, Algèbre et Calcul* (LAC) of

---

\(^{82}\)See [http://www.lri.fr/~conchon/gerardmer/](http://www.lri.fr/~conchon/gerardmer/).


\(^{84}\)See [http://www.cs.bham.ac.uk/~drg/galop08.html](http://www.cs.bham.ac.uk/~drg/galop08.html).

\(^{85}\)See [http://www.floc-conference.org/LSB-home.html](http://www.floc-conference.org/LSB-home.html).


\(^{87}\)See [http://pauillac.inria.fr/~herbelin/aipa2012/](http://pauillac.inria.fr/~herbelin/aipa2012/).

the CNRS GDR\textsuperscript{89} Informatique Mathématique (GDR-IM): one in February 2007 (Chambéry, 44 attendees) and one in November 2010 (Paris, 53 attendees)\textsuperscript{90}. She also organized the PPS lab yearly September meeting in Trouville, in 2011 (first venue) and 2012.

- Jean Krivine organized the \textit{Bertinoro Summer School on Natural Computing} in June 2008, a lecture series on Modeling bio molecular systems and the \textit{Santa Fe Institute Complex System Summer School} in 2008 a lecture series on Rule based modeling.

- Melliès initiated with Nick Benton (Microsoft Research) in 2010 the LOLA workshop devoted to the syntax and semantics of low level languages. The workshop has been organized since then (LOLA 2010, LOLA 2011, LOLA 2012) as a satellite workshop of the LICS conference. He has also organized a series of thematic meetings in the PPS lab, devoted to operad theory (2009, 2010), synthetic geometry (2010), categorification (2011), homotopy type theory (2012)


- Régis-Gianas organized a joint workshop between the Eternal INRIA ARC project and the CerCo European project (around 25 participants).

- Saurin organized an RDP workshop in 2011: the first \textit{Workshop on the Theory and Practice of Delimited Control} took place in Novi Sad during the RDP 2011 week. Taking the occasion of the physical PC meeting of ICFP 2011, Saurin organized an informal workshop involving PC members, opened to all members of the PPS lab.

- Sozeau organised a one-day workshop on \textit{Reification and generic tactics} at INRIA, Paris in March 2011, with 20 participants\textsuperscript{91}. He was a coorganiser of the \textit{DTP’11} workshop, August 2011, Nijmegen, Netherlands\textsuperscript{92}.

\textsuperscript{89}A GDR (\textit{Groupe de recherche}) is a CNRS research structure which gathers members of various laboratories on specific research topics. They have a low CNRS specific funding but are extremely successful and efficient for federating research communities.


\textsuperscript{91}See http://coq.inria.fr/cocorico/GTReification/.

\textsuperscript{92}See http://www.cs.ru.nl/dtp11/.
• Di Cosmo and Treinen organised a two day workshop on Software Reliability at IRILL, Paris, in June 2011, with over 60 participants from academia and industry\textsuperscript{93}.

• Varacca was one of the creators and organizers of the regular CHOCO Seminar, a one-day meeting taking place on an (almost) monthly basis at LIP, ENS Lyon and organized by the CHOCO ANR Blanc project (lead by Ehrhard). Started in 2007, these meetings gathered members from the project – IML (Université d’Aix-Marseille 2), LIF (Université d’Aix-Marseille 1), LIP (ENS Lyon), LAMA (Université de Chambéry), LIPN (Université Paris Nord – Villetteuse) and PPS – and had actually a wider audience\textsuperscript{94}. After the end of the CHOCO project, these successful events have been followed on by the CHoCoLa\textsuperscript{95} series, still coorganized by Varacca.

References


\textsuperscript{93}See \url{http://www.irill.org/events/journees-irill-gtll-sur-la-fiabilite-du-logiciel/}.

\textsuperscript{94}See \url{http://choco.pps.jussieu.fr/events/}.

\textsuperscript{95}See \url{http://chocola.ens-lyon.fr/}.


<table>
<thead>
<tr>
<th>Ref</th>
<th>Reference</th>
</tr>
</thead>
</table>


[Can11] Benjamin Canou. Programmation Web Typée. These, Université Pierre et Marie Curie - Paris VI, October 2011. 41


110


Stéphane Glondu. *Vers une certification de l’extraction de Coq*. PhD thesis, Université Paris Diderot, June 2012. 21


<table>
<thead>
<tr>
<th>Reference</th>
<th>Author/Title</th>
<th>Details</th>
</tr>
</thead>
</table>