UMR IRISA

# Activity Report 2025

## Team EPICURE

Semantic analysis and compilation for secure execution environments

*Joint team with Centre Inria de l'Université de Rennes*

D4 – Language and Software Engineering

Université de Rennes | rennes | INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES | cnrs | Inria | Inserm | CentraleSupélec | IMT Atlantique Bretagne-Pays de la Loire École Mines-Télécom | Université Bretagne Sud ubs

# Project-Team EPICURE

*Creation of the Project-Team: 2022 June 01*

Each year, Inria research teams publish an Activity Report presenting their work and results over the reporting period. These reports follow a common structure, with some optional sections depending on the specific team. They typically begin by outlining the overall objectives and research programme, including the main research themes, goals, and methodological approaches. They also describe the application domains targeted by the team, highlighting the scientific or societal contexts in which their work is situated. The reports then present the highlights of the year, covering major scientific achievements, software developments, or teaching contributions. When relevant, they include sections on software, platforms, and open data, detailing the tools developed and how they are shared. A substantial part is dedicated to new results, where scientific contributions are described in detail, often with subsections specifying participants and associated keywords. Finally, the Activity Report addresses funding, contracts, partnerships, and collaborations at various levels, from industrial agreements to international cooperations. It also covers dissemination and teaching activities, such as participation in scientific events, outreach, and supervision. The document concludes with a presentation of scientific production, including major publications and those produced during the year.

# Keywords

## Computer sciences and digital sciences

A2.1. – Programming Languages

A2.1.1. – Semantics of programming languages

A2.1.2. – Imperative programming

A2.1.3. – Object-oriented programming

A2.1.4. – Functional programming

A2.2. – Compilation

A2.2.1. – Static analysis

A2.2.5. – Run-time systems

A2.2.9. – Security by compilation

A4. – Security and privacy

A4.4. – Security of equipment and software

A4.5. – Formal method for verification, reliability, certification

A4.5.1. – Static analysis

A4.5.3. – Program proof

## Other research topics and application domains

B6.1.1. – Software engineering

B6.4. – Internet of things

B6.6. – Embedded systems

# Contents

**Project-Team EPICURE** **1**

**1 Team members, visitors, external collaborators** **5**

**2 Overall objectives** **6**

**3 Research program** **6**

**4 Application domains** **7**
4.1 Internet of Things . . . 7
4.2 High-assurance blockchains . . . 7

**5 Social and environmental responsibility** **8**

**6 Highlights of the year** **8**

**7 Latest software developments, platforms, open data** **8**
7.1 Latest software developments . . . 8
7.1.1 necro . . . 8
7.1.2 Timbuk . . . 8
7.1.3 dmap . . . 9
7.1.4 sexp_decode . . . 9
7.1.5 CompcertSSA . . . 9
7.1.6 plantinator . . . 9
7.1.7 Salto Static Analyser . . . 10
7.2 Open data . . . 10

**8 New results** **10**
8.1 Skeletal Semantics . . . 10
8.2 Verified Compilation . . . 10
8.3 Contextual equivalence saturation . . . 11
8.4 Back to the trees: Identifying plants with Human intelligence . . . 11
8.5 Non-Deterministic Abstract Machines as Semantic Models . . . 12
8.6 Proving satisfiability of CHCs using Answer Set Programming . . . 12

**9 Bilateral contracts and grants with industry** **12**
9.1 Bilateral contracts with industry . . . 12

**10 Partnerships and cooperations** **13**
10.1 International research visitors . . . 13
10.1.1 Visits of international scientists . . . 13
10.2 National initiatives . . . 13
10.2.1 PEPR Cybersécurité Secureval . . . 13
10.2.2 Action Exploratoire "Back to the Trees" and ANR SAPS Flores . . . 14

**11 Dissemination** **14**
11.1 Promoting scientific activities . . . 14
11.1.1 Scientific events: organisation . . . 14
11.1.2 Scientific events: selection . . . 14
11.1.3 Journal . . . 15
11.1.4 Invited talks . . . 15
11.1.5 Scientific expertise . . . 15
11.1.6 Research administration . . . 15
11.2 Teaching - Supervision - Juries - Educational and pedagogical outreach . . . 16
11.2.1 Supervision . . . 16

11.2.2 Juries . . . 17
11.3 Popularization . . . 17
11.3.1 Participation in Live events . . . 17
11.3.2 Others science outreach relevant activities . . . 17

**12 Scientific production** **18**
12.1 Major publications . . . 18
12.2 Publications of the year . . . 19
12.3 Cited publications . . . 20

# 1 Team members, visitors, external collaborators

## Research Scientists

- Thomas Jensen [Team leader, INRIA, Senior Researcher, HDR]
- Frédéric Besson [INRIA, Researcher]
- Simon Castellan [INRIA, Researcher]
- Benoit Montagu [INRIA, Researcher]
- Alan Schmitt [INRIA, Senior Researcher, HDR]

## Faculty Members

- Sandrine Blazy [UNIV RENNES, Professor, HDR]
- Delphine Demange [UNIV RENNES, Associate Professor]
- Benjamin Farinier [UNIV RENNES, Associate Professor]
- Thomas Genet [UNIV RENNES, Professor, HDR]

## PhD Students

- Martin Andrieux [UNIV RENNES, from Sep 2025]
- Sebastien Bonduelle [INRIA]
- Clement Chavanon [INRIA]
- Alexandre Drewery [INRIA]
- Sophia La Spina [UNIV RENNES, until Aug 2025]
- Tony Law [INRIA, from Nov 2025]
- Tony Law [UNIV RENNES, until Sep 2025]
- Malo Revel [UNIV RENNES]
- Yann Salmon [UNIV RENNES, from Oct 2025]

## Technical Staff

- Gurvan Cabon [INRIA, Engineer]
- Pierre Lermusiaux [INRIA, Engineer, until Aug 2025]

## Interns and Apprentices

- Martin Andrieux [ENS RENNES, Intern, until Jul 2025]
- Baptiste Izquierdo Rey [ENS DE LYON, Intern, from Jun 2025 until Jul 2025]
- Timothee Meneux [INRIA, Intern, from May 2025 until Jul 2025]
- Yann Salmon [INRIA, Intern, from Apr 2025 until Sep 2025]
- Alex Tual–Hamon [INRIA, Intern, from May 2025 until Aug 2025]

### Administrative Assistant

- Lydie Mabil [INRIA]

### Visiting Scientists

- Sean Kristian Remond Harbo [UNIV AALBORG, from Oct 2025 until Nov 2025]
- Sean Kristian Remond Harbo [UNIV AALBORG, from Mar 2025 until Apr 2025]

## 2 Overall objectives

The security of the software that surrounds us is, more than ever, a scientific challenge of utmost societal importance. More and more software is produced to operate on an increasingly varied number of devices and to provide increasingly complex functionality. There is a pressing need to provide the science and technology for engineering software so that it becomes safe and secure, in addition to providing the desired functionality. This need is not new and a multitude of programming languages, semantic theories, formal methods, verification tools and techniques have been developed and contribute to meet this need. One of the challenges with this state of affairs is exactly the multitude of languages in which to express the algorithms that we develop, and in particular the distance between those languages for which it is comparatively easy to develop correct and secure software, and those that actually get executed in our computers, telephones, pacemakers, cars, smart home IoT devices *etc.*.

No one single silver bullet will solve the problem of developing secure software worthy of the user's trust. We are however convinced that a cornerstone of the answer is *programming language semantics*, *i.e.*, a mathematically robust yet flexible formalism for defining the behaviour of a program. The goal of the EPICURE project is to contribute with semantics-based methods for producing safe and secure software by

- defining new semantic frameworks that will provide more accurate models of modern execution platforms, and which can facilitate the semantic definition of the above-mentioned multitude of programming languages,
- designing formally verified analysis and compilation schemes, with the specific aim of being able to analyse and verify properties of programs written in high-level languages, and to compile both program and the verified properties down to low-level executable representations,
- demonstrate the impact of language-based tools on software security by showing how they can improve the correctness, safety and security of critical software found in modern execution environments, such as the Java virtual machine, the Tezos blockchain written in OCaml, and small operating systems for the IoT such as RIOT.

## 3 Research program

The overall goal of the EPICURE project is to guarantee the security and safety of key software components of execution platforms, including those used in the IoT and blockchains. Our contribution to this goal will be to develop semantics-based, formally verifiable program analyses and compilation techniques for improving and enforcing software security and safety. The main open challenges in the field include:

- providing mechanised formalisations of modern programming languages (such as Rust, JavaScript, Web Assembly) which facilitate the reasoning about these languages and their tools,
- faithfully modeling architectures on which they execute, taking into account features such as out-of-order execution and trust-enhancing mechanisms such as enclaves and trust zones,
- designing program processing tools such as analyses and compilers, the correctness of which can be verified mechanically,

- developing scalable analyses for proving security properties of high-level programs, and compiling programs and their proofs down to low-level executables, the security of which is guaranteed by the compilation process.

The EPICURE project is structured into the following research axes:

- Semantics and their mechanisation.
- Program analysis.
- Trustworthy compilation.
- Secure execution platforms.

The axis on semantics and their mechanisation will investigate frameworks for defining semantics, in particular the recently proposed *skeletal* semantics and the notion of causal semantics. We will pay particular attention to the semantics of intermediate representations used in compilers and to the semantic description of low-level languages, *e.g.* eBPF. In the axis on program analysis, we plan to conduct work both on the foundations of static analysis and abstract interpretation and on the development of specific analyses, in particular for higher-order polymorphic functional programs. A special attention will be given to the problem of translating results of an analysis from a high-level language to its compiled (low-level) version. In the strand on trustworthy compilation we will pursue the effort on mechanised verification of optimising compilers. We will also examine the security impact of compilation with respect to different (passive and active) attacker models. The intended application areas for these techniques are the Internet of Things and high-assurance block chains.

# 4 Application domains

The intended application of the scientific results outlined in the previous sections is to improve the safety and security of execution platforms, taken in a broad sense ranging from virtual machines to hardware processors. We will improve on analyses and compilation techniques for verifying and producing safer code, as we will improve on the key software tools and components that implement the execution platform. In this section we outline a number of more concrete applications that we intend to investigate.

## 4.1 Internet of Things

The Internet of Things offers a large and diverse domain of application for our formal methods. The limitations of the devices populating the IoT mean that a different kind of algorithms are deployed but the security and privacy concerns remain, and are even accentuated by the relative weak protection mechanisms offered by the underlying hardware. In particular, the IoT relies on cryptographic primitives for secure communication and software updates but these primitives are often different from what is used on standard execution platforms due to the limited computing resources. The question of secure compilation and the techniques that we expect to develop can be transferred to the IoT but the security properties might be harder to verify because of optimisations.

On the application level, the distributed and asynchronous nature of the IoT has led to new programming paradigms and novel uses of existing languages (such as JavaScript) that pose new verification challenges, in particular the verification of coordinating programs written in different complex languages in a multitier framework. A multitier language unifies within a single formalism and a single execution environment the programming of the different tiers of distributed applications. On the web, this paradigm unifies the client tier, the server tier, and the database tier. We thus want to investigate how our techniques can be brought to bear on multitier programming languages. In particular, we propose to investigate the design of program analyses for a multitier language for the IoT.

## 4.2 High-assurance blockchains

Because they enable the distributed management of virtual assets—such as property rights, proofs of payments— blockchain systems play a growing, *critical* role in our societies. Blockchain-based systems, like

Ethereum or Tezos, are equipped with so-called *contracts*. A contract is a program which is executed by a *virtual machine* (VM) over the blockchain. The effect of a contract is to update values and assets stored in the blockchain. Thus, any failure in the safety, availability, or security in the VM of a system like Tezos could have dramatic consequences on industries, on public infrastructures, and eventually on people. The pieces of code that lie at the foundations of the Tezos system are entrusted with the safety and security of all the managed assets. The Tezos core software is thus expected to attain the highest levels of clarity and quality, and to get as close as possible to zero defects. This is where formal methods—and in particular static analyses—can help, by giving guarantees about the dynamic behaviour of programs, in an automatic way. The expressive type system of OCaml—the implementation language of Tezos—already provides static safety guarantees by ensuring data is used in a consistent way. In collaboration with Nomadic Labs, we will provide OCaml programs with additional guarantees, by answering questions such as *"can a program raise an exception?"*, *"can a program break some user-defined invariant?"*, or *"which data might be modified by a program?"*. Those questions are beyond the scope of the OCaml type system, but are within reach of abstract interpretation-based static analyses. The endeavour of supporting all the features of OCaml is beyond the scope of this project. Instead, we will target a representative subset of the pure fragment of the OCaml language, in which the core of Tezos's VM is written.

# 5 Social and environmental responsibility

EPICURE runs the INRIA exploratory action "Back to the trees" which aims to use probabilistic programming and Bayesian inference to produce a plant identification tool that is reliable, educational and convivial, built together with botanist collectives.

# 6 Highlights of the year

The Epicure team organised the 53rd ACM SIGPLAN Symposium on Principles of Programming Languages (POPL 2026) together with several co-located conferences which all took place in Rennes.

# 7 Latest software developments, platforms, open data

## 7.1 Latest software developments

### 7.1.1 necro

**Name:** necro

**Keywords:** Semantics, Programming language, Specification language

**Functional Description:** The goal of the project is to provide a tool to manipulate skeletal semantics, a format to represent the semantics of programming languages. This tool has been mostly developed by Victoire Noizet.

**URL:** http://skeletons.inria.fr/necro.html

**Publication:** tel-03855276v1

**Contact:** Alan Schmitt

**Participant:** Alan Schmitt

### 7.1.2 Timbuk

**Keywords:** Automated deduction, Ocaml, Program verification, Tree Automata, Term Rewriting Systems

**Functional Description:** Timbuk is a tool designed to compute or over-approximate sets of terms reachable by a given term rewriting system. The library also provides an OCaml top-level with all usual functions on Bottom-up Nondeterministic Tree Automata.

**URL:** http://people.irisa.fr/Thomas.Genet/timbuk/index.html

**Contact:** Thomas Genet

**Participant:** Thomas Genet

#### 7.1.3 dmap

**Name:** dependent maps library in OCaml

**Keywords:** Ocaml, Library, Data structures

**Functional Description:** dmap is an OCaml library that implements immutable maps, for which the type of data may depend on the key they are associated with.

**URL:** https://gitlab.inria.fr/bmontagu/dmap

**Contact:** Benoit Montagu

**Participant:** Benoit Montagu

#### 7.1.4 sexp_decode

**Keywords:** Ocaml, Library

**Functional Description:** sexp_decode is an OCaml library of monadic combinators for decoding S-expressions (as defined in the Csexp library) into structured data.

**URL:** https://gitlab.inria.fr/bmontagu/sexp_decode

**Contact:** Benoit Montagu

**Participant:** Benoit Montagu

#### 7.1.5 CompcertSSA

**Keywords:** Optimizing compiler, Formal methods, Proof assistant, SSA

**Functional Description:** CompcertSSA is built on top of the Compcert verified C compiler, by adding a middle-end based on the SSA form (Static Single Assignment) : conversion to SSA, SSA-based optimizations, and destruction of SSA.

**URL:** https://compcertssa.gitlabpages.inria.fr/

**Publications:** hal-01378393, hal-01193281, hal-02904204, hal-03899435, hal-01110783, hal-01097677, hal-01110779

**Contact:** Delphine Demange

**Participants:** Delphine Demange, Sandrine Blazy, David Pichardie, 2 anonymous participants

#### 7.1.6 plantinator

**Name:** plantinator

**Keywords:** Data management, Algebraic Data Types, Decision

**Functional Description:** Plantinator is a database management software for morphological data about plants as well as automatic identification key generator

**URL:** https://botascopia.inria.fr

**Contact:** Simon Castellan

**Participants:** Simon Castellan, Eric Tannier, Gurvan Cabon

#### 7.1.7 Salto Static Analyser

**Keywords:** Static analysis, Ocaml, Abstract interpretation

**Scientific Description:** Static analyser for OCaml programs, that supports recursive algebraic data types (including GADT and non regular types), first-class functions, first-class exceptions, dynamic exceptions, first-class modules, mutable data types (mutable records, arrays), and base types such as integers, floating point numbers, characters, and strings.

The analyser infers for every program point an abstract value that represents an over-approximation of the set of values that this program point can compute, and of the exceptions that can be raised.

**Functional Description:** Detection of uncaught exceptions, possible exit codes, undefined behaviours in OCaml programs. This static analyser is based on the theory of abstract interpretation.

**News of the Year:** Extension of the scope of features supported by the analyser, and support for a large part of the OCaml standard library. Analysis of whole projects that are built using the dune build system.

**URL:** https://salto.gitlabpages.inria.fr/

**Publications:** hal-04547480, hal-04410771, hal-04769799

**Contact:** Benoit Montagu

**Participants:** Benoit Montagu, Pierre Lermusiaux, Thomas Jensen, Thomas Genet

### 7.2 Open data

## 8 New results

### 8.1 Skeletal Semantics

**Participants:** Martin Andrieux, , Thomas Jensen, , Alan Schmitt.

The work on skeletal semantics [29], a modular and formal way to describe semantics or programming languages, has continued during 2025. Links to papers and tools can be found at the dedicated website.

Martin Andrieux did a Master internship (M2) on the the transformation of non-algebraic effects into algebraic effects. The goal of his work, continued in the setting of his PhD started in September 2025, is to decouple the effects of the semantics of a language from its description, paving the way toward more efficient compilation and a shallow embedding into Rocq.

### 8.2 Verified Compilation

**Participants:** Sandrine Blazy, Delphine Demange, Tony Law, Sophia La Spina, Clement Chavanon.

In 2025, we continued our work on verified compilation using the Rocq proof assistant, focusing on formalizing dataflow solvers, new intermediate representations for dataflow circuits and a domain specific language for packet filtering.

We published our work on the formalization of specific dataflow solvers inspired by the work of Bourdoncle. There, an iteration order is pre-computed, based on the structure of the control-flow graph of programs. Central to the proof of correctness is the general notion of a Weak Topological Ordering (WTO). Our correctness proofs are valid for any such ordering. The first solver implements an iterative strategy over the ordering, the second solver implements a recursive strategy. Our solvers are extractable to OCaml code. Our formalization is fully compatible with the interface of dataflow solvers within the verified, optimizing

C CompCert compiler. We conducted practical experiments on the wide range of forward and backward analyses from CompCert, demonstrating the practicality of our solvers in terms of efficiency and precision. A research article [20] was accepted for publication at ESOP 2025, that describes our formalization and experiments. Sophia La Spina continued her PhD research work and formalized in Rocq an efficient algorithm to compute a satifying WTO for any control-flow graph. The proof is conducted in Rocq, and integrates with both WTO-based dataflow solvers. Its performance are comparable to a state-of-the-art, non-verified implementation in OCaml from the OCamlGraph library. The details of the proof are presented in Sophia La Spina's PhD manuscript [27].

New results on dataflow circuits propose a mechanized formal semantics: rather than following a static schedule predetermined at generation time, the execution of the components in a circuit is constrained solely by the availability of their input data. Circuit components are modeled as abstract computing units, asynchronously connected with each other through unidirectional, unbounded FIFO. We formalize sufficient conditions to achieve the determinacy of circuits executions: all possible schedules of such circuits lead to a unique observable behavior. Moreover, we provide two equivalent views for circuits. The first one is a direct and natural representation as graphs of components. The second is a core, structured term calculus, which enables constructing and reasoning about circuits in a inductive way. We prove that both representations are semantically equivalent. We experimentally validate its relevance by applying our general semantic framework to dataflow circuits generated with Dynamatic, a recent HLS tool exploiting dataflow circuits to generate dynamically scheduled, elastic circuits. A research article [21] was accepted for publication at OOPSLA 2025, that describes our formalization and experiments. In 2025, Tony Law continued his PhD work on formalizing circuits at a lower level of abstraction. Essentially, the proposed semantics is making explicit the communication protocol used by elastic circuit components to exchange data tokens. This lower-level semantics is formally proven to refine the higher-level dataflow circuit semantics, provided that adequate, formal, criteria hold on the circuit components. These criteria provably hold on the set of components used in our reference Dynamatic HLS compiler producing elastic circuits.

New results on mechanized semantics using the Rocq proof assistant propose a different style of writing and mechanizing semantics that follows the CEK abstract machine. We explain how this style interacts better with basic proof automation tactics than the traditional structural operational semantics style for the $\lambda$-calculus. In [18], we explicitly detail why this alternative semantic style allows for scaling up more efficiently the mechanized proof development, and hint at future improvements based on the same principles. After detailing the inner workings of the interactions between the semantic style and proof automation of common theorems, we perform a case study for the medium-sized Catala domain-specific programming language. This is a joint work with Alain Delaët and Denis Merigoux.

## 8.3 Contextual equivalence saturation

**Participants:** Thomas Jensen, Alexandre Drewery.

Equality saturation is a semantics-based technique for automatically and efficiently proving that two programs are equivalent modulo a fixed set of equality axioms. The technique works by first converting both programs into a term representation of their semantics, called a Program Expression Graph. Such PEGs can be loaded in an e-graph, a data structure originally developed for representing equivalence classes of terms in SMT solvers. Equality saturation then consists in applying a series of rewriting rules on e-graphs in order to prove that the two programs have equivalent semantics.

In this work, we have extended the equality saturation technique with contextual reasoning in order to perform rewriting under assumptions that are locally valid inside a conditional branch. This is based on a new notion of cyclic e-graphs with contextual annotations. We experimentally validate the efficiency and scalability of this new technique by proving equivalence of several families of programs where contextual reasoning is required. The work was presented at the 32nd Static Analysis Symposium [19] and was carried out in collaboration with David Pichardie (Meta).

## 8.4 Back to the trees: Identifying plants with Human intelligence

**Participant:** Simon Castellan, Gurvan Cabon, Eric Tannier.

Plant descriptions have been recorded by botanists since a few thousands years. Identifying the characteristic criteria of a species, (invariants under all the individuals of that species), as well as describing precise morphology with words, proved to be a difficult endeavour. As a result, each botanist tend to use their own vocabulary and way of describing species. Moreover descriptions tend not to be uniform. We show how to use basic tools from type theory and probabilistic programming can help account for morphological diversity in a way that can be understood by a machine.

By providing a compositional metalanguage to describe traits as types, we empower botanists to work together to create a probabilistic representation of plants that accounts for diverse form of polymorphism in plants. Each description becomes a probabilistic element of that type. This data can then be turned automatically into identification keys to help people learn plant identification.

New results showing how to use probabilistic programming to convert between different type, to turn a scientific description into an informal one are being investigated.

This system led to the interdisciplinary publication [25].

### 8.5 Non-Deterministic Abstract Machines as Semantic Models

**Participant:** Małgorzata Biernacka, Dariusz Biernacki, Sergueï Lenget, Alan Schmitt.

In complement to skeletal semantics, we explore the definition and formalization of the semantics of programming languages through the specification of non-deterministic abstract machines and the study of their properties.

In 2025, we worked on defining and relating two serialized semantics format for process calculi based on labeled transition systems. Complementary semantics are useful to prove the congruence of bisimilarity. Zipper semantics are the first step toward the automatic derivation of an abstract machine. Both semantics are based on the same restrictions: inductive rules have at most one premise and some contextual information is kept in the label. This work has been presented in [23].

### 8.6 Proving satisfiability of CHCs using Answer Set Programming

**Participant:** Thomas Genet.

For formal verification of programs, Constrained Horn Clauses (CHCs) are commonly used as an intermediate representation. Programs are translated into a set of positive CHCs and the properties as negative CHCs. If the global set of CHCs is satisfiable (if it has a model) then the property is true. Thus, most verification techniques on CHCs explicitly build such a model. When programs manipulate recursive Algebraic Data Types, models are Herbrand models. Those models are generally unbounded because they range over terms defined by recursive data types (e.g. unbounded lists). However, such models can be finitely represented using tree automata. We proposed a relatively complete technique to infer such a finite representation based on Answer Set Programming. The technique is *relatively* complete because it terminates only if the Herbrand model is regular (i.e. it can be recognized by a a tree automaton). Preliminary results were presented at the 12th Workshop on Horn Clauses for Verification and Synthesis (HCVS) [28].

## 9 Bilateral contracts and grants with industry

### 9.1 Bilateral contracts with industry

**Development of Salto**

**Participants:** Pierre Lermusiaux, Benoît Montagu.

**Title:** Salto

**Industrial partner:** OCaml Software Foundation

**Date/Duration:** one year (Nov 2024 – Oct 2025)

**Participants:** Pierre Lermusiaux, Benoît Montagu

**Additional info/keywords:** The OCaml Software Foundation supported the development of the Salto static analyser thanks to a one year grant, to fund a research engineer. The goal is to broaden the scope of the analyser, improve its precision and its efficiency, and experimenting with new ideas, so that the analyser can be released to the OCaml community in the near future.

# 10 Partnerships and cooperations

## 10.1 International research visitors

### 10.1.1 Visits of international scientists

**Other international visits to the team**

**Kristian Harbo**

PhD student

**Institution of origin:** University of Aalborg

**Country:** Denmark

**Dates:** March and November 2025.

**Context of the visit:** Development of a skeletal semantics for WebAssembly

%subsubsectionVisits to international teams

## 10.2 National initiatives

### 10.2.1 PEPR Cybersécurité Secureval

**Participants:** Thomas Jensen, Frederic Besson, Sandrine Blazy, Benjamin Farinier, Alexandre Drewery, Clement Chavanon, Benoit Montagu, Sebastien Bonduelle.

The Secureval project concerns the assessment of the security of digital systems. Digital system security assessment relies on compliance and vulnerability analyses to provide recognized cybersecurity assurances. Innovative tools will be designed around new digital technologies in order to verify the absence of hardware and software vulnerabilities, and to carry out the required proof of conformity. EPICURE contributes with research on advanced static analysis and verified compilation techniques.

#### 10.2.2 Action Exploratoire "Back to the Trees" and ANR SAPS Flores

**Participants:** Simon Castellan, Gurvan Cabon.

The Botascopia project aims at gather morphological and ecological structured data about flowering plants in order to make available tools for botanical outreach under the form of identification keys. The project was awarded an *Action Exploratoire Inria* in March 2023 to explore the use of methods from programming languages theory to represent faithfully botanical data as well as permit a bridge between technical description and informal descriptions used for the general public.

The project was later awarded an ANR *Science Avec et Pour la Société* (from January 2024 to December 2025) in partnerships with botanists and a national citizen science association, Tela-Botanica to develop a common platform to enter and transform botanical data into identification keys. The project is led with botanists from Université Paris-Saclay, Université de Rennes, CNRS, and computer scientists from Inria Lyon.

Both the ANR project and the Action exploratoire ended at the end of 2025.

# 11 Dissemination

## 11.1 Promoting scientific activities

### 11.1.1 Scientific events: organisation

- Sandrine Blazy: Steering Committe of POPL
- Sandrine Blazy: Steering Committe of CPP
- Sandrine Blazy: Steering Committe of ITP
- Alan Schmitt: Steering Committee of JFLA
- Benoit Montagu: Steering Committe of POPL
- Benoit Montagu: Steering Committe of ML Workshop

**General chair, scientific chair**

- Sandrine Blazy, 53rd ACM SIGPLAN Symposium on Principles of Programming Languages *POPL 2026*, General Chair
- Alan Schmitt, *CORSE - Components Operationally: Reversibility and System Engineering*, 2025, co-chair

**Member of the organizing committees**

- Alan Schmitt, *POPL 2026*, Workshops and Tutorials co-chair
- Benoit Montagu, *POPL 2026*, Industrial Relations co-chair
- Benoit Montagu, *ICFP 2025*, Artifact Evaluation co-chair

### 11.1.2 Scientific events: selection

**Chair of conference program committees**

- Sandrine Blazy, *CPP 2025*, PC co-chair
- Thomas Jensen: 27th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI 2026)

**Member of the conference program committees**

- Sandrine Blazy: program committee member of ICFP 2025
- Sandrine Blazy: program committee member of PriSC 2025
- Alan Schmitt, *FORTE 2025*
- Benoit Montagu, VMCAI 2026
- Benoit Montagu, JFLA 2026

**Reviewer**

- Alan Schmitt, *ICFP 2025*
- Benoit Montagu, ICFP 2025

#### 11.1.3 Journal

- Thomas Jensen ACM Trans. on Programming Languages and Systems.
- Thomas Genet Journal of Functional Programming.

**Member of the editorial boards**

- Sandrine Blazy: member of the editorial board of the LMCS journal

**Reviewer - reviewing activities**

- Alan Schmitt, *LMCS*, Special issue for *LICS*
- Benoit Montagu, Frontiers in Computer Science

#### 11.1.4 Invited talks

- Sandrine Blazy: Talk on verified compilation at the mentoring workshop PLMW 2025 co-locatezd with POPL 2025
- Sandrine Blazy: Talk on verified compilation, Dagstuhl seminar on WebAssembly, June 2025
- Sandrine Blazy: Keynote talk, GDR Sécurité informatique, Caen, June 2025
- Sandrine Blazy: Keynote talk, 37th Euromicro Conference on Real-Time Systems (ECRTS 2025), Brussels, Belgium, July 2025
- Sandrine Blazy: Keynote talk, Brazilian symposium on programming languages (SBLP), Recife, Brazil, September 2025
- Thomas Jensen: Program analysis for software security, COPLAW'26, Copenhagen, Denmark.
- Thomas Jensen: Course on software security, Summer school, Vrije U. Bruxelles, Belgium.

#### 11.1.5 Scientific expertise

- Alan Schmitt, Member of Conseil Scientifique of LMF, Formal Methods Laboratory, Paris Saclay

#### 11.1.6 Research administration

- Sandrine Blazy is deputy director of the IRISA CNRS laboratory.
- Thomas Jensen is director of the Laboratoire d'excellence CominLabs.

## 11.2 Teaching - Supervision - Juries - Educational and pedagogical outreach

- Licence : Sandrine Blazy, Programmation de Confiance, 55h, L3, Université Rennes, France
- Master : Sandrine Blazy, Mechanized Semantics, 32h, M1, Université Rennes, France
- Licence: Alan Schmitt, L3 INFO, 18h, ENS Rennes, France
- Master: Alan Schmitt, Advanced Semantics, 32h, M2, ENS Rennes, France
- Master: Alan Schmitt, Preparation of Agregation exam, 30h, M2, ENS Rennes, France
- Master: Simon Castellan, Informatique et anthropocène, M2, 12h, ENS Rennes, France
- Master: Benoit Montagu, Software Security, 6h, M2, Master SIF Rennes, France
- Licence: Frédéric Besson, Introduction to functional programming, 30h, L3, Insa Rennes, France
- Master : Thomas Jensen, Software Security, 12h, M2, Master SIF Rennes, France.
- Master : Thomas Jensen, Software Security, 20h, Master, University of Copenhagen, Denmark
- Licence : Thomas Genet, Algorithmique et complexité, 40h, L1, Université de Rennes, France
- Licence : Thomas Genet, Sécurité des logiciels et des protocoles, 30h, L3, Université de Rennes, France
- Master : Thomas Genet, Analyse et Conception Formelles, 38h, M1, Université Rennes, France
- Master : Thomas Genet, Blockchains, 12h, M2, Université Rennes, France
- Master : Thomas Genet, Blockchains, 9h, M2, IMT Atlantique, France
- Licence : Delphine Demange, Introduction to imperative programming, 42h, L1, Université de Rennes, France
- Licence : Delphine Demange, Algorithmics and Experimental Complexity, 36h, L1, Université de Rennes, France
- Licence : Delphine Demange, Trustworthy programming in Why3, 34h, L3, Université de Rennes, France
- Licence : Delphine Demange, Introduction to Compilation, 42h, L3, Université de Rennes, France
- Master : Delphine Demange, Abstract Syntax-directed Programming for Compilers, 48h, M1, Université de Rennes, France

### 11.2.1 Supervision

- L3 internship, Baptiste Izquierdo Rey : Simon Castellan
- M1 internship, Timothée Meneux: Alan Schmitt
- M1 internship, Alex Tual–Hamon: Sandrine Blazy
- M2 internship, Martin Andrieux: Alan Schmitt
- PhD (ongoing), Martin Andrieux, "Compilation Générique Certifiée par Machines Abstraites": Alan Schmitt
- PhD (ongoing), Sebastien Bonduelle, "Information Flow Static Analyses by Abstract Interpretation": Thomas Jensen, Benoit Montagu
- PhD (ongoing), Clement Chavanon, Refinement of formal specifications for secure environments" since September 2023: Sandrine Blazy, Frédéric Besson

- PhD (ongoing), Yann Salmon, "Formal Verification of Root of Trust Software", since October 2025: Thomas Jensen, Frédéric Besson
- PhD (ongoing), Alexandre Drewery, "Proving Program equivalence with e-graphs", since October 2023: Thomas Jensen
- Research engineer, Pierre Lermusiaux, Salto project: Benoit Montagu
- PhD (defended Dec. 16th 2025), Sophia La Spina, "Formal Verification of Weak Topological Orderings and Dataflow Analyses", 2022-2025: Sandrine Blazy, Delphine Demange
- PhD (ongoing), Tony Law, "Mechanised semantics for circuits: from data flow to hardware", since October 2022: Sandrine Blazy, Delphine Demange
- PhD in progress, Malo Revel: “Proving regular theorems on functional programs”, Thomas Genet and Thomas Jensen.

#### 11.2.2 Juries

- Thomas Jensen : rapporteur on the PhD thesis "Precise and Efficient Memory Analysis of Low-level languages” by Julien Simonnet, Université Paris-Saclay.
- Sandrine Blazy: jury member (president) for the PhD defense of Clément Allain, University Paris Cité, December 2025.
- Sandrine Blazy: jury member for the PhD defense of Antoine Geimer, Rennes University, December 2025.
- Sandrine Blazy: jury member (president) for the PhD defense of Nicolas Bailluet, Rennes University, November 2025.
- Sandrine Blazy: jury member for the PhD defense of Josué Moreau, University Paris Saclay, November 2025.
- Sandrine Blazy: jury member (reviewer) for the PhD defense of Maxime Legoupil, Aarhus University, Denmark, November 2025.
- Sandrine Blazy: jury member (president) for the PhD defense of Adrienne Lancelot, Institut Polytechnique de Paris, November 2025.
- Sandrine Blazy: jury member (president) for the HDR defense of Caterina Urban, University Paris Sciences et Lettres, September 2025.
- Delphine Demange : jury member (examiner) for PhD defense of Antonin Reitz, University Paris Sciences et Lettres, December 2025.

### 11.3 Popularization

#### 11.3.1 Participation in Live events

- Delphine Demange: "JuraSTIC Exhibition", a general public exhibition about computer-science and computers history, organized as part of the Fête de la Science, and the 50th Anniversary of IRISA. Collaboration with Service Culturel et Collections of University of Rennes. Le Diapason, Rennes, France.

#### 11.3.2 Others science outreach relevant activities

- Thomas Genet: "Bug, virus, pirates. So many threats and no solution? Yes, mathematics.", given in 4 High Schools close to Rennes.
- Delphine Demange : "Programming Languages, Semantics, and Formal Software Verification", for computer-science, informatics and electronics students of ISTIC, Rennes, France.

# 12 Scientific production

## 12.1 Major publications


[1] O. Andreescu, T. Jensen, S. Lescuyer and B. Montagu. 'Inferring frame conditions with static correlation analysis'. In: *Proceedings of the ACM on Programming Languages* 3.POPL (2nd Jan. 2019), pp. 1–29. DOI: 10.1145/3290360. URL: https://hal.inria.fr/hal-02413262.

[2] A. Barrière, S. Blazy, O. Flückiger, D. Pichardie and J. Vitek. 'Formally verified speculation and deoptimization in a JIT compiler'. In: *Proceedings of the ACM on Programming Languages* 5.POPL (4th Jan. 2021), p. 26. DOI: 10.1145/3434327. URL: https://hal.science/hal-03185848.

[3] A. Barrière, S. Blazy and D. Pichardie. 'Formally Verified Native Code Generation in an Effectful JIT - or: Turning the CompCert Backend into a Formally Verified JIT Compiler'. In: *Proceedings of the ACM on Programming Languages* (Jan. 2023). DOI: 10.1145/3571202. URL: https://hal.inria.fr/hal-03882598.

[4] G. Barthe, S. Blazy, B. Grégoire, R. Hutin, V. Laporte, D. Pichardie and A. Trieu. 'Formal verification of a constant-time preserving C compiler'. In: *Proceedings of the ACM on Programming Languages* 4.POPL (Jan. 2020), pp. 1–30. DOI: 10.1145/3371075. URL: https://hal.univ-lorraine.fr/hal-02975012.

[5] F. Besson, S. Blazy, A. Dang, T. Jensen and P. Wilke. 'Compiling Sandboxes: Formally Verified Software Fault Isolation'. In: ESOP 2019 - 28th European Symposium on Programming. Vol. 11423. LNCS. Prague, Czech Republic: Springer, 6th Apr. 2019, pp. 499–524. DOI: 10.1007/978-3-030-17184-1_18. URL: https://hal.inria.fr/hal-02316189.

[6] F. Besson, A. Dang and T. Jensen. 'Information-Flow Preservation in Compiler Optimisations'. In: CSF 2019 - 32nd IEEE Computer Security Foundations Symposium. Hoboken, United States: IEEE, 25th June 2019, pp. 1–13. URL: https://hal.inria.fr/hal-02180303.

[7] M. Bodin, A. Charguéraud, D. Filaretti, P. Gardner, S. Maffeis, D. Naudziuniene, A. Schmitt and G. Smith. 'A Trusted Mechanised JavaScript Specification'. In: POPL 2014 - 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. San Diego, United States, 22nd Jan. 2014. URL: https://hal.inria.fr/hal-00910135.

[8] M. Bodin, P. Gardner, T. Jensen and A. Schmitt. 'Skeletal Semantics and their Interpretations'. In: *Proceedings of the ACM on Programming Languages* 44 (2019), pp. 1–31. DOI: 10.1145/3290357. URL: https://hal.inria.fr/hal-01881863.

[9] S. Castellan and P. Clairambault. 'The Geometry of Causality: Multi-Token Geometry of Interaction and its Causal Unfolding'. In: *Proceedings of the ACM on Programming Languages* (Jan. 2023). URL: https://hal.science/hal-03286443.

[10] T. Haudebourg, T. Genet and T. Jensen. 'Regular Language Type Inference with Term Rewriting - extended version'. In: *Proceedings of the ACM on Programming Languages*. International Conference on Functional Programming (ICFP) 4.ICFP (2020), pp. 1–29. DOI: 10.1145/3408994. URL: https://hal.inria.fr/hal-02795484.

[11] J.-H. Jourdan, V. Laporte, S. Blazy, X. Leroy and D. Pichardie. 'A formally-verified C static analyzer'. In: POPL 2015: 42nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. Mumbai, India: ACM, 15th Jan. 2015, pp. 247–259. DOI: 10.1145/2676726.2676966. URL: https://hal.inria.fr/hal-01078386.

[12] R. La Spina, D. Demange and S. Blazy. 'Formal Verification of WTO-based Dataflow Solvers'. In: *Programming Languages and Systems*. ESOP 2025 - 34th European Symposium on Programming. Hamilton, Canada, 2025, pp. 1–27. URL: https://hal.science/hal-04851724.

[13] T. Law, D. Demange and S. Blazy. 'A Mechanized Semantics for Dataflow Circuits'. In: *Proceedings of the ACM on Programming Languages, Issue OOPSLA 1*. SPLASH 2025 - ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity. Singapore, Singapore: ACM, 2025, pp. 1–29. URL: https://hal.science/hal-04851772.

[14] P. Lermusiaux and B. Montagu. 'Detection of Uncaught Exceptions in Functional Programs by Abstract Interpretation'. In: *Programming Languages and Systems, 33rd European Symposium on Programming, ESOP 2024, Lecture Notes in Computer Science*. ESOP 2024 - 33rd European Symposium on Programming. Vol. 14577. Lecture Notes in Computer Science. Luxembourg, Luxembourg, 5th Apr. 2024, pp. 391–420. DOI: 10.1007/978-3-031-57267-8_15. URL: https://hal.science/hal-04547480.

[15] B. Montagu and T. Jensen. 'Stable relations and abstract interpretation of higher-order programs'. In: *Proceedings of the ACM on Programming Languages* 4.ICFP (2nd Aug. 2020), pp. 1–30. DOI: 10.1145/3409001. URL: https://hal.inria.fr/hal-02916996.

[16] B. Montagu and T. Jensen. 'Trace-Based Control-Flow Analysis'. In: PLDI 2021 - 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation. Virtual, Canada: ACM, 20th June 2021, pp. 1–15. DOI: 10.1145/3453483.3454057. URL: https://hal.inria.fr/hal-03266981.

## 12.2 Publications of the year

### International peer-reviewed conferences

[17] M. Andrieux and A. Schmitt. 'Algebraizing Higher-Order Effects'. In: JFLA 2026 – 37es Journées Francophones des Langages Applicatifs. Vol. JFLA 2026 – 37es Journées Francophones des Langages Applicatifs. Oberbronn, Alsace, France, 2026. URL: https://hal.science/hal-05428145.

[18] A. Delaët, S. Blazy and D. Merigoux. 'Abstract Machines and Small-step Semantics: a Winning Ticket for Proof Automation?' In: PPDP 2025 - 27th International Symposium on Principles and Practice of Declarative Programming. Rende, Italy, 10th Sept. 2025. URL: https://inria.hal.science/hal-05471781 (cit. on p. 11).

[19] A. Drewery, T. Jensen and D. Pichardie. 'Contextual Equality Saturation'. In: SAS 2025 - 32nd Static Analysis Symposium. Singapore, Singapore, 2025, pp. 1–26. URL: https://inria.hal.science/hal-05226543 (cit. on p. 11).

[20] R. La Spina, D. Demange and S. Blazy. 'Formal Verification of WTO-based Dataflow Solvers'. In: *Programming Languages and Systems*. ESOP 2025 - 34th European Symposium on Programming. Hamilton, Canada, 2025, pp. 1–27. URL: https://hal.science/hal-04851724 (cit. on p. 11).

[21] T. Law, D. Demange and S. Blazy. 'A Mechanized Semantics for Dataflow Circuits'. In: *Proceedings of the ACM on Programming Languages, Issue OOPSLA 1*. SPLASH 2025 - ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity. Singapore, Singapore: ACM, 2025, pp. 1–29. URL: https://hal.science/hal-04851772 (cit. on p. 11).

### Scientific books

[22] C. A. Mezzina and A. Schmitt, eds. *Components Operationally: Reversibility and System Engineering: Essays Dedicated to Jean-Bernard Stefani on the Occasion of His 65th Birthday*. Vol. 16065. Lecture Notes in Computer Science. Springer, 2026. DOI: 10.1007/978-3-031-99717-4. URL: https://inria.hal.science/hal-05325655.

### Scientific book chapters

[23] S. Lenglet, C. Noûs and A. Schmitt. 'From Complementary to Zipper Semantics'. In: *Components Operationally: Reversibility and System EngineeringEssays Dedicated to Jean-Bernard Stefani on the Occasion of His 65th Birthday*. Vol. 16065. Lecture Notes in Computer Science. Springer Nature Switzerland, 19th Oct. 2025, pp. 89–102. DOI: 10.1007/978-3-031-99717-4_5. URL: https://inria.hal.science/hal-05322293 (cit. on p. 12).

### Reports & preprints

[24] Y. Boichut and T. Genet. *Checking that Equations on Terms define a Finite Set of Equivalence Classes*. 2025. URL: https://hal.science/hal-05117243.

[25] S. Castellan, A. Alcolei, A. Atlan, X. Aubriot, M. Bellot, G. Cabot, L. Dahmani, J. Käfer, S. Nadot, S. Oort, A. Schermann-Legionnet and E. Tannier. *Botascopia: a digital setup for generating low tech plant field guides*. 2025. URL: https://hal.science/hal-05419216 (cit. on p. 12).

[26] M. Revel, T. Genet and T. Jensen. *Complete Abstractions for Verification of Polymorphic Functions with Equality – extended version*. IRISA, 12th Jan. 2026. URL: https://inria.hal.science/hal-05454439.

### Other scientific publications

[27] S. La Spina. *Formal verification of weak topological orderings and dataflow analysis*. 16th Dec. 2025. URL: https://inria.hal.science/hal-05471604 (cit. on p. 11).

### Scientific popularization

[28] G. Maire and T. Genet. 'Finding Regular Herbrand Models for CHCs using Answer Set Programming'. In: HCVS 2025 - 12th Workshop on Horn Clauses for Verification and Synthesis. Zagreb, Croatia, 22nd July 2025. URL: https://inria.hal.science/hal-05459509 (cit. on p. 12).

## 12.3 Cited publications

[29] L. Noizet and A. Schmitt. 'Semantics in Skel and Necro'. In: *ICTCS 2022 - Italian Conference on Theoretical Computer Science*. CEUR Workshop Proceedings. Rome, Italy, Sept. 2022, pp. 1–17. URL: https://inria.hal.science/hal-03784478 (cit. on p. 10).