RESEARCH CENTRE

Inria Centre at Université Grenoble Alpes

IN PARTNERSHIP WITH:

Université de Grenoble Alpes

2024

ACTIVITY REPORT

Project-Team

CONVECS

Construction of verified concurrent systems

IN COLLABORATION WITH: Laboratoire d'Informatique de Grenoble (LIG)

DOMAIN

Algorithmics, Programming, Software and Architecture

THEME

Proofs and Verification

Inria

Contents

Project-Team CONVECS

Creation of the Project-Team: 2014 January 01

Keywords

Computer sciences and digital sciences

A1.3.3. – Blockchain

A1.3.5. – Cloud

A2.1.1. – Semantics of programming languages

A2.1.6. – Concurrent programming

A2.1.7. – Distributed programming

A2.4.1. – Analysis

A2.4.2. – Model-checking

A2.5. – Software engineering

A2.5.1. – Software Architecture & Design

A2.5.4. – Software Maintenance & Evolution

A2.5.5. – Software testing

A5.10.1. – Design

A6.1.3. – Discrete Modeling (multi-agent, people centered)

A7.1.1. – Distributed algorithms

A7.1.3. – Graph algorithms

A7.2. – Logic in Computer Science

A8.9. – Performance evaluation

Other research topics and application domains

B5.1. – Factory of the future

B5.4. – Microelectronics

B5.6. – Robotic systems

B6.1.1. – Software engineering

B6.3.2. – Network protocols

B6.4. – Internet of things

B6.5. – Information systems

B6.6. – Embedded systems

B7.2.1. – Smart vehicles

1 Team members, visitors, external collaborators

Research Scientists

- Radu Mateescu [Team leader, Inria, Senior Researcher]
- Hubert Garavel [Inria, Senior Researcher]
- Frederic Lang [Inria, Researcher]
- Wendelin Serwe [Inria, Researcher]

Faculty Member

- Gwen Salaün [UGA, Professor]

Post-Doctoral Fellows

- Wei Chen [Inria, Post-Doctoral Fellow, from Sep 2024]
- Aline Uwimbabazi [SCHLUMBERGER]

PhD Students

- Zachary Assoumani [Inria, from Oct 2024]
- Irman Faqrizal [UGA]
- Suraj Gupta [Euris]
- Ahmed Khebbeb [UGA, from Dec 2024]
- Philippe Ledent [Inria, until Sep 2024]
- Quentin Nivon [UGA]
- Ahang Zuo [UGA, ATER, until Aug 2024]

Technical Staff

- Abdelilah Mejdoubi [Inria, Engineer, from Nov 2024]

Interns and Apprentices

- Andres Diaz Hernandez [Inria, Intern, from Mar 2024 until Apr 2024]

Administrative Assistant

- Myriam Etienne [Inria]

External Collaborator

- Pierre Boullier [retired Inria senior researcher]

2 Overall objectives

2.1 Overview

The CONVECS project-team addresses the rigorous design of concurrent asynchronous systems using formal methods and automated analysis. These systems comprise several activities that execute simultaneously and autonomously (i.e., without the assumption about the existence of a global clock), synchronize, and communicate to accomplish a common task. In computer science, asynchronous concurrency arises typically in hardware, software, and telecommunication systems, but also in parallel and distributed programs.

Asynchronous concurrency is becoming ubiquitous, from the micro-scale of embedded systems (asynchronous logic, networks-on-chip, GALS – *Globally Asynchronous, Locally Synchronous* systems, multi-core processors, etc.) to the macro-scale of grids and cloud computing. In the race for improved performance and lower power consumption, computer manufacturers are moving towards asynchrony. This increases the complexity of the design by introducing nondeterminism, thus requiring a rigorous methodology, based on formal methods assisted by analysis and verification tools.

There exist several approaches to formal verification, such as theorem proving, static analysis, and model checking, with various degrees of automation. When dealing with asynchronous systems involving complex data types, verification methods based on state space exploration (reachability analysis, model checking, equivalence checking, etc.) are today the most successful way to detect design errors that could not be found otherwise. However, these verification methods have several limitations: they are not easily accepted by industry engineers, they do not scale well while the complexity of designs is ever increasing, and they require considerable computing power (both storage capacity and execution speed). These are the challenges that CONVECS seeks to address.

To achieve significant impact in the design and analysis of concurrent asynchronous systems, several research topics must be addressed simultaneously. There is a need for user-friendly, intuitive, yet formal specification languages that will be attractive to designers and engineers. These languages should provide for both functional aspects (as needed by formal verification) and quantitative ones (to enable performance evaluation and architecture exploration). These languages and their associated tools should be smoothly integrated into large-scale design flows. Finally, verification tools should be able to exploit the parallel and distributed computing facilities that are now ubiquitous, from desktop to high-performance computers.

3 Research program

3.1 New Formal Languages and their Concurrent Implementations

We aim at proposing and implementing new formal languages for the specification, implementation, and verification of concurrent systems. In order to provide a complete, coherent methodological framework, two research directions must be addressed:

- *Model-based specifications*: these are operational (i.e., constructive) descriptions of systems, usually expressed in terms of processes that execute concurrently, synchronize together and communicate. Process calculi are typical examples of model-based specification languages. The approach we promote is based on LOTOS NT (LNT for short), a formal specification language that incorporates most constructs stemming from classical programming languages, which eases its acceptance by students and industry engineers. LNT [6] is derived from the ISO standard E-LOTOS (2001), of which it represents the first successful implementation, based on a source-level translation from LNT to the former ISO standard LOTOS (1989). We are working both on the semantic foundations of LNT (enhancing the language with module interfaces and timed/probabilistic/stochastic features, compiling the *m* among *n* synchronization, etc.) and on the generation of efficient parallel and distributed code. Once equipped with these features, LNT will enable formally verified asynchronous concurrent designs to be implemented automatically.

- *Property-based specifications*: these are declarative (i.e., non-constructive) descriptions of systems, which express *what* a system should do rather than *how* the system should do it. Temporal logics

and μ-calculi are typical examples of property-based specification languages. The natural models underlying value-passing specification languages, such as LNT, are Labeled Transition Systems (LTSs or simply *graphs*) in which the transitions between states are labeled by actions containing data values exchanged during handshake communications. In order to reason accurately about these LTSs, temporal logics involving data values are necessary. The approach we promote is based on MCL (*Model Checking Language*) [44], which extends the modal μ-calculus with data-handling primitives, fairness operators encoding generalized Büchi automata, and a functional-like language for describing complex transition sequences. We are working both on the semantic foundations of MCL (extending the language with new temporal and hybrid operators, translating these operators into lower-level formalisms, enhancing the type system, etc.) and also on improving the MCL on-the-fly model checking technology (devising new algorithms, enhancing ergonomy by detecting and reporting vacuity, etc.).

We address these two directions simultaneously, yet in a coherent manner, with a particular focus on applicable concurrent code generation and computer-aided verification.

3.2 Parallel and Distributed Verification

Exploiting large-scale high-performance computers is a promising way to augment the capabilities of formal verification. The underlying problems are far from trivial, making the correct design, implementation, fine-tuning, and benchmarking of parallel and distributed verification algorithms long-term and difficult activities. Sequential verification algorithms cannot be reused as such for this task: they are inherently complex, and their existing implementations reflect several years of optimizations and enhancements. To obtain good speedup and scalability, it is necessary to invent new parallel and distributed algorithms rather than to attempt a parallelization of existing sequential ones. We seek to achieve this objective by working along two directions:

- *Rigorous design:* Because of their high complexity, concurrent verification algorithms should themselves be subject to formal modeling and verification, as confirmed by recent trends in the certification of safety-critical applications. To facilitate the development of new parallel and distributed verification algorithms, we promote a rigorous approach based on formal methods and verification. Such algorithms will be first specified formally in LNT, then validated using existing model checking algorithms of the CADP toolbox. Second, parallel or distributed implementations of these algorithms will be generated automatically from the LNT specifications, enabling them to be experimented on large computing infrastructures, such as clusters and grids. As a side-effect, this "bootstrapping" approach would produce new verification tools that can later be used to self-verify their own design.

- *Performance optimization:* In devising parallel and distributed verification algorithms, particular care must be taken to optimize performance. These algorithms will face concurrency issues at several levels: grids of heterogeneous clusters (architecture-independence of data, dynamic load balancing), clusters of homogeneous machines connected by a network (message-passing communication, detection of stable states), and multi-core machines (shared-memory communication, thread synchronization). We will seek to exploit the results achieved in the parallel and distributed computing field to improve performance when using thousands of machines by reducing the number of connections and the messages exchanged between the cooperating processes carrying out the verification task. Another important issue is the generalization of existing LTS representations (explicit, implicit, distributed) in order to make them fully interoperable, such that compilers and verification tools can handle these models transparently.

3.3 Timed, Probabilistic, and Stochastic Extensions

Concurrent systems can be analyzed from a *qualitative* point of view, to check whether certain properties of interest (e.g., safety, liveness, fairness, etc.) are satisfied. This is the role of functional verification, which produces Boolean (yes/no) verdicts. However, it is often useful to analyze such systems from a *quantitative* point of view, to answer non-functional questions regarding performance over the long run,

response time, throughput, latency, failure probability, etc. Such questions, which call for numerical (rather than binary) answers, are essential when studying the performance and dependability (e.g., availability, reliability, etc.) of complex systems.

Traditionally, qualitative and quantitative analyzes are performed separately, using different modeling languages and different software tools, often by distinct persons. Unifying these separate processes to form a seamless design flow with common modeling languages and analysis tools is therefore desirable, for both scientific and economic reasons. Technically, the existing modeling languages for concurrent systems need to be enriched with new features for describing quantitative aspects, such as probabilities, weights, and time. Such extensions have been well-studied and, for each of these directions, there exist various kinds of automata, e.g., discrete-time Markov chains for probabilities, weighted automata for weights, timed automata for hard real-time, continuous-time Markov chains for soft real-time with exponential distributions, etc. Nowadays, the next scientific challenge is to combine these individual extensions altogether to provide even more expressive models suitable for advanced applications.

Many such combinations have been proposed in the literature, and there is a large amount of models adding probabilities, weights, and/or time. However, an unfortunate consequence of this diversity is the confuse landscape of software tools supporting such models. Dozens of tools have been developed to implement theoretical ideas about probabilities, weights, and time in concurrent systems. Unfortunately, these tools do not interoperate smoothly, due both to incompatibilities in the underlying semantic models and to the lack of common exchange formats.

To address these issues, CONVECS follows two research directions:

- *Unifying the semantic models.* Firstly, we will perform a systematic survey of the existing semantic models in order to distinguish between their essential and non-essential characteristics, the goal being to propose a unified semantic model that is compatible with process calculi techniques for specifying and verifying concurrent systems. There are already proposals for unification either theoretical (e.g., Markov automata) or practical (e.g., PRISM and MODEST modeling languages), but these languages focus on quantitative aspects and do not provide high-level control structures and data handling features (as LNT does, for instance). Work is therefore needed to unify process calculi and quantitative models, still retaining the benefits of both worlds.

- *Increasing the interoperability of analysis tools.* Secondly, we will seek to enhance the interoperability of existing tools for timed, probabilistic, and stochastic systems. Based on scientific exchanges with developers of advanced tools for quantitative analysis, we plan to evolve the CADP toolbox as follows: extending its perimeter of functional verification with quantitative aspects; enabling deeper connections with external analysis components for probabilistic, stochastic, and timed models; and introducing architectural principles for the design and integration of future tools, our long-term goal being the construction of a European collaborative platform encompassing both functional and non-functional analyzes.

3.4 Component-Based Architectures for On-the-Fly Verification

On-the-fly verification fights against state explosion by enabling an incremental, demand-driven exploration of LTSs, thus avoiding their entire construction prior to verification. In this approach, LTS models are handled implicitly by means of their *post* function, which computes the transitions going out of given states and thus serves as a basis for any forward exploration algorithm. On-the-fly verification tools are complex software artifacts, which must be designed as modularly as possible to enhance their robustness, reduce their development effort, and facilitate their evolution. To achieve such a modular framework, we undertake research in several directions:

- *New interfaces for on-the-fly LTS manipulation.* The current application programming interface (API) for on-the-fly graph manipulation, named OPEN/CAESAR [31], provides an “opaque” representation of states and actions (transitions labels): states are represented as memory areas of fixed size and actions are character strings. Although appropriate to the pure process algebraic setting, this representation must be generalized to provide additional information supporting an efficient construction of advanced verification features, such as: handling of the types, functions, data

values, and parallel structure of the source program under verification, independence of transitions in the LTS, quantitative (timed/probabilistic/stochastic) information, etc.

- *Compositional framework for on-the-fly LTS analysis.* On-the-fly model checkers and equivalence checkers usually perform several operations on graph models (LTSs, Boolean graphs, etc.), such as exploration, parallel composition, partial order reduction, encoding of model checking and equivalence checking in terms of Boolean equation systems, resolution and diagnostic generation for Boolean equation systems, etc. To facilitate the design, implementation, and usage of these functionalities, it is necessary to encapsulate them in software components that could be freely combined and replaced. Such components would act as graph transformers, that would execute (on a sequential machine) in a way similar to coroutines and to the composition of lazy functions in functional programming languages. Besides its obvious benefits in modularity, such a component-based architecture will also make it possible to take advantage of multi-core processors.

- *New generic components for on-the-fly verification.* The quest for new on-the-fly components for LTS analysis must be pursued, with the goal of obtaining a rich catalog of interoperable components serving as building blocks for new analysis features. A long-term goal of this approach is to provide an increasingly large catalog of interoperable components covering all verification and analysis functionalities that appear to be useful in practice. It is worth noticing that some components can be very complex pieces of software (e.g., the encapsulation of an on-the-fly model checker for a rich temporal logic). Ideally, it should be possible to build a novel verification or analysis tool by assembling on-the-fly graph manipulation components taken from the catalog. This would provide a flexible means of building new verification and analysis tools by reusing generic, interoperable model manipulation components.

3.5 Real-Life Applications and Case Studies

We believe that theoretical studies and tool developments must be confronted with significant case studies to assess their applicability and to identify new research directions. Therefore, we seek to apply our languages, models, and tools for specifying and verifying formally real-life applications, often in the context of industrial collaborations.

4 Application domains

The theoretical framework we use (automata, process algebras, bisimulations, temporal logics, etc.) and the software tools we develop are general enough to fit the needs of many application domains. They are applicable to virtually any system or protocol that consists of distributed agents communicating by asynchronous messages. The list of recent case studies performed with the CADP toolbox (see in particular § 6.5) illustrates the diversity of applications:

- *Bioinformatics:* genetic regulatory networks, nutritional stress response, metabolic pathways,
- *Component-based systems:* Web services, peer-to-peer networks,
- *Cloud computing:* self-deployment protocols, dynamic reconfiguration protocols,
- *Databases:* transaction protocols, distributed knowledge bases, stock management,
- *Distributed systems:* virtual shared memory, dynamic reconfiguration algorithms, fault tolerance algorithms, multi-agent systems,
- *Embedded systems:* air traffic control, autonomous vehicles, avionic systems, train supervision systems, medical devices,
- *Enterprise systems:* business processes, information systems, manufacturing,
- *Fog and IoT:* stateful IoT applications in the fog, industrial IoT,

- *Hardware architectures:* multiprocessor architectures, systems on chip, cache coherency protocols, hardware/software codesign,
- *Human-machine interaction:* graphical interfaces, biomedical data visualization, plasticity,
- *Security protocols:* authentication, electronic transactions, cryptographic key distribution,
- *Telecommunications:* high-speed networks, network management, mobile telephony, feature interaction detection.

5 New software, platforms, open data

5.1 New software

5.1.1 CADP

Name: Construction and Analysis of Distributed Processes

Keywords: Formal methods, Verification

Functional Description: CADP (*Construction and Analysis of Distributed Processes* – formerly known as *CAESAR/ALDEBARAN Development Package*) [5] is a toolbox for protocols and distributed systems engineering.

In this toolbox, we develop and maintain the following tools:

- CAESAR.ADT [30] is a compiler that translates LOTOS abstract data types into C types and C functions. The translation involves pattern-matching compiling techniques and automatic recognition of usual types (integers, enumerations, tuples, etc.), which are implemented optimally.
- CAESAR [36, 35] is a compiler that translates LOTOS processes into either C code (for rapid prototyping and testing purposes) or finite graphs (for verification purposes). The translation is done using several intermediate steps, among which the construction of a Petri net extended with typed variables, data handling features, and atomic transitions.
- OPEN/CAESAR [31] is a generic software environment for developing tools that explore graphs on the fly (for instance, simulation, verification, and test generation tools). Such tools can be developed independently of any particular high level language. In this respect, OPEN/CAESAR plays a central role in CADP by connecting language-oriented tools with model-oriented tools. OPEN/CAESAR consists of a set of 16 code libraries with their programming interfaces, such as:
  - CAESAR_GRAPH, which provides the programming interface for graph exploration,
  - CAESAR_HASH, which contains several hash functions,
  - CAESAR_SOLVE, which resolves Boolean equation systems on the fly,
  - CAESAR_STACK, which implements stacks for depth-first search exploration, and
  - CAESAR_TABLE, which handles tables of states, transitions, labels, etc.

  A number of on-the-fly analysis tools have been developed within the OPEN/CAESAR environment, among which:
  - BISIMULATOR, which checks bisimulation equivalences and preorders,
  - CUNCTATOR, which performs steady-state simulation of continuous-time Markov chains,
  - DETERMINATOR, which eliminates stochastic nondeterminism in normal, probabilistic, or stochastic systems,
  - DISTRIBUTOR, which generates the graph of reachable states using several machines,
  - EVALUATOR, which evaluates MCL formulas,

  - EXECUTOR, which performs random execution,
  - EXHIBITOR, which searches for execution sequences matching a given regular expression,
  - GENERATOR, which constructs the graph of reachable states,
  - PROJECTOR, which computes abstractions of communicating systems,
  - REDUCTOR, which constructs and minimizes the graph of reachable states modulo various equivalence relations,
  - SIMULATOR, XSIMULATOR, and OCIS, which enable interactive simulation, and
  - TERMINATOR, which searches for deadlock states.

- BCG (*Binary Coded Graphs*) is both a file format for storing very large graphs on disk (using efficient compression techniques) and a software environment for handling this format. BCG also plays a key role in CADP as many tools rely on this format for their inputs/outputs. The BCG environment consists of various libraries with their programming interfaces, and of several tools, such as:
  - BCG_CMP, which compares two graphs,
  - BCG_DRAW, which builds a two-dimensional view of a graph,
  - BCG_EDIT, which allows the graph layout produced by BCG_DRAW to be modified interactively,
  - BCG_GRAPH, which generates various forms of practically useful graphs,
  - BCG_INFO, which displays various statistical information about a graph,
  - BCG_IO, which performs conversions between BCG and many other graph formats,
  - BCG_LABELS, which hides and/or renames (using regular expressions) the transition labels of a graph,
  - BCG_MIN, which minimizes a graph modulo strong or branching equivalences (and can also deal with probabilistic and stochastic systems),
  - BCG_STEADY, which performs steady-state numerical analysis of (extended) continuous-time Markov chains,
  - BCG_TRANSIENT, which performs transient numerical analysis of (extended) continuous-time Markov chains, and
  - XTL (*eXecutable Temporal Language*), which is a high level, functional language for programming exploration algorithms on BCG graphs. XTL provides primitives to handle states, transitions, labels, *successor* and *predecessor* functions, etc.
    For instance, one can define recursive functions on sets of states, which allow evaluation and diagnostic generation fixed point algorithms for usual temporal logics (such as HML [37], CTL[27], ACTL[28], etc.) to be defined in XTL.

- PBG (*Partitioned BCG Graph*) is a file format implementing the theoretical concept of *Partitioned LTS* [34] and providing a unified access to a graph partitioned in fragments distributed over a set of remote machines, possibly located in different countries. The PBG format is supported by several tools, such as:
  - PBG_CP, PBG_MV, and PBG_RM, which facilitate standard operations (copying, moving, and removing) on PBG files, maintaining consistency during these operations,
  - PBG_MERGE (formerly known as BCG_MERGE), which transforms a distributed graph into a monolithic one represented in BCG format,
  - PBG_INFO, which displays various statistical information about a distributed graph.

- The connection between explicit models (such as BCG graphs) and implicit models (explored on the fly) is ensured by OPEN/CAESAR-compliant compilers, e.g.:
  - BCG_OPEN, for models represented as BCG graphs,
  - CAESAR.OPEN, for models expressed as LOTOS descriptions,
  - EXP.OPEN, for models expressed as communicating automata,

- FSP.OPEN, for models expressed as FSP [41] descriptions,
- LNT.OPEN, for models expressed as LNT descriptions, and
- SEQ.OPEN, for models represented as sets of execution traces.

The CADP toolbox also includes TGV (*Test Generation based on Verification*), which has been developed by the VERIMAG laboratory (Grenoble) and Inria Rennes – Bretagne-Atlantique.

The CADP tools are well-integrated and can be accessed easily using either the EUCALYPTUS graphical interface or the SVL [32] scripting language. Both EUCALYPTUS and SVL provide users with an easy and uniform access to the CADP tools by performing file format conversions automatically whenever needed and by supplying appropriate command-line options as the tools are invoked.

URL: http://cadp.inria.fr/

Contact: Hubert Garavel

Participants: Hubert Garavel, Frederic Lang, Radu Mateescu, Wendelin Serwe

5.1.2 TRAIAN

Keywords: Compilation, LOTOS NT

Functional Description: TRAIAN is a compiler for translating LOTOS NT descriptions into C programs, which will be used for simulation, rapid prototyping, verification, and testing.

The current version of TRAIAN, which handles LOTOS NT types and functions only, has useful applications in compiler construction [33], being used in all recent compilers developed by CONVECS.

URL: http://convecs.inria.fr/software/traian/

Contact: Hubert Garavel

Participants: Hubert Garavel, Frederic Lang, Wendelin Serwe

6 New results

6.1 New Formal Languages and their Implementations

6.1.1 LNT and LOTOS NT Specification Languages

Participants: Hubert Garavel, Frédéric Lang, Wendelin Serwe.

LNT [6] [26] is a next-generation formal description language for asynchronous concurrent systems. The design of LNT at CONVECS is the continuation of the efforts undertaken in the 80s to define sound languages for concurrency theory and, indeed, LNT is derived from the ISO standards LOTOS (1989) and E-LOTOS (2001). In a nutshell, LNT attempts to combine the best features of imperative programming languages, functional languages, and value-passing process calculi.

LNT is not a frozen language: its definition started in 2005, as part of an industrial project. Since 2010, LNT has been systematically used by CONVECS for numerous case studies (many of which being industrial applications — see § 6.5). LNT is also used as a back-end by other research teams who implement various languages by translation to LNT. It is taught in university courses, e.g., at University Grenoble Alpes and ENSIMAG, where it is positively accepted by students and industry engineers. Based on the feedback acquired by CONVECS, LNT is continuously improved.

LOTOS NT is a language predecessor of LNT, equipped with the TRAIAN compiler, which is used for the construction of most CADP compilers and translators. Since TRAIAN 3.0, the lexer and parser of TRAIAN are built using the SYNTAX compiler-generation system developed at Inria Paris and the

abstract syntax tree of LOTOS NT, the library of predefined LOTOS NT types and functions, the static semantics checking (identifier binding, type checking, dataflow analysis, etc.), and the C code generation are implemented in LOTOS NT itself, so that TRAIAN is capable of bootstrapping itself.

In 2024, our efforts led to the release of four new major versions of TRAIAN, which introduce various changes to the LOTOS NT language in order to further align it with LNT:

- TRAIAN 3.13 brings a new syntax for the "select" and "trap" constructs, as well as enhancements to static analysis and code generation.
- TRAIAN 3.14 focuses on static analysis and increased convergence with LNT2LOTOS, as the semantic checks of LNT2LOTOS are progressively migrated to TRAIAN.
- TRAIAN 3.15 pursues the progressive migration to TRAIAN of the semantic checks formerly done by LNT2LOTOS. Following a systematic review and comparison with the checks implemented in LNT2LOTOS, the error and warning messages displayed by TRAIAN are now more complete and informative.
- TRAIAN 3.16 is a consolidation release that completes the migration to TRAIAN of all static semantic checks formerly implemented in LNT2LOTOS. This release also introduces the concept of virtual processes that enable generic modules parameterized by processes.

Each version of TRAIAN includes various compiler and code-generation changes accomodating the newly introduced features. For each version, the “traian_upc” conversion tool was extended to ease the upgrade of existing LOTOS NT source code whenever possible, and the user manual of TRAIAN, as well as the demo examples, were constantly updated.

The LNT language continued its evolution to become a better language and to achieve convergence between LNT and the LOTOS NT language supported by the TRAIAN compiler. In 2024, the LNT2LOTOS and TRAIAN compilers have been progressively aligned with each other, leading to changes in the definition of LNT:

- The “select” keyword was renamed to “alt” (as a tribute to Tony Hoare).
- LNT2LOTOS was enhanced to support “library” directives for including LNT files, definitions of synonym types, clauses “with nil” and “with cons” in lists and sets, clauses “with diff” and “with minus” in sorted lists and sets, virtual processes (i.e., imported processes that are neither defined in the current module, nor in its transitively included modules).
- Patterns of the form “P where V” can now appear everywhere a pattern is accepted, and LNT2LOTOS implements a large subset of them.
- A predefined channel EXIT was introduced for LNT events that model exceptions, in particular for the UNEXPECTED event, and channel definitions with the “raise” attribute are now supported by LNT2LOTOS.
- The ambiguity of expressions of the form “X [Y]” is now resolved using semantic information (either X is an array and Y an index, or X is a function and Y an event).

The tighter compatibility between TRAIAN and LNT2LOTOS, and the fact that TRAIAN is now systematically invoked before LNT2LOTOS, enabled simplifications and code factorization between the various tools for LNT:

- Many messages formerly emitted by LNT2LOTOS have been changed (165 error messages and 23 warning messages also emitted by TRAIAN have been suppressed, and 18 error messages now display as “limitations”, to indicate that the corresponding problems are due to advanced features that LNT2LOTOS does not handle yet). The removal of these static-semantics checks reduced by 10% the number of lines of code in LNT2LOTOS, making it noticeably faster on correct LNT programs.

- The LNT_CHECK and LNT_DEPEND auxiliary tools were removed from CADP, being now subsumed by TRAIAN. Also the LPP preprocessor was suppressed, since the translation to LOTOS of LNT constants (natural numbers, reals, characters, and strings) is now done by LNT2LOTOS instead of LPP. Consequently, the "-lnt" option of LOTOS.OPEN, the "-pidlist" and "-notraian" options of LNT.OPEN, and the "-depend", "-pidlist", and "-traian" options of LNT2LOTOS have been removed. These changes significantly improved the speed of the LNT tool chain, the time needed by LNT.OPEN to compile 13,500+ correct LNT programs being reduced from 25 hours down to 11 hours 40 minutes.

Twelve bugs have been fixed in the various tools for LNT. New verifications and better error messages have been added to LNT_MERGE and LNT.OPEN. The following algorithmic enhancements have been brought to LNT2LOTOS:

- The algorithm for inlining simple functions in the generated LOTOS code was revised, dividing by 15 the execution time of LNT2LOTOS on certain large examples.
- LNT2LOTOS no longer generates dead LOTOS code when translating "B1; B2" if it detects that all the execution paths in B1 lead to deadlocks, infinite loops, breaks of an enclosing loop, or raises of events that are not trapped.
- LNT2LOTOS no longer generates extra LOTOS code for checking that each event properly matches its channel (as this check is done by TRAIAN), and now generates modified C code to avoid raising the GCC 12 warnings about infinite recursion.

The LNT2LOTOS Reference Manual, the LNT tools, the EUCALYPTUS user interface, the editor style files for LNT, the CADP manual pages, many CADP demos, and the web sites for CADP and TRAIAN have been updated to implement and document the changes described above. In particular, a manual page for TRAIAN was added to CADP and there is now one single entry point gathering all information available about LNT. The "upc" shell script was extended to ease the migration of LNT programs.

6.1.2 Nested-Unit Petri Nets

Participants: Hubert Garavel.

Nested-Unit Petri Nets (NUPNs) is a model of computation that can be seen as an upward-compatible extension of P/T nets, which are enriched with structural information on their concurrent and hierarchical structure. Such structural information can easily be produced when NUPNs are generated from higher-level specifications (e.g., process calculi) and allows logarithmic reductions in the number of bits required to represent reachable states, thus enabling verification tools to perform better. For this reason, NUPNs have been so far implemented in thirteen verification tools developed in four countries, and adopted by two international competitions (the Model Checking Contest and the Rigorous Examination of Reactive Systems challenge). The complete theory of NUPNs is formalized in a journal article [3] and their PNML representation is described here.

In 2024, in collaboration with Pierre Bouvier (Kalray), we proposed efficient techniques for detecting isomorphism between nets, i.e., for identifying, in large collections of (safe) Petri nets or NUPNs, all the nets that are identical modulo a permutation of places, a permutation of transitions, and/or a permutation of units. Our approach relies upon the successive application of diverse algorithms of increasing complexities: net signatures, net canonizations, and characterization of isomorphic nets in terms of isomorphic graphs. We implemented this approach in a complete tool chain that we successfully assessed on four collections, the largest of which comprises 241,000+ nets with many duplicates. This work led to a publication in an international conference [12].

6.1.3 Formal Modeling and Analysis of BPMN

Participants: Quentin Nivon, Gwen Salaün *(correspondent)*.

Modelling and designing business processes has become a crucial activity for companies in the last 20 years. As a consequence, multiple workflow modelling notations were proposed. BPMN (*Business Process Modelling Notation*) is one of them and is now considered as the de facto standard for process modelling.

The BPMN notation offers a rich syntax that requires a certain level of expertise before being able to write correct and well-structured processes corresponding to some expected requirements. The BPMN modelling phase can thus be tedious and error-prone if carried out by non-experts. The main goal of the approach presented in this paper is to help users modelling BPMN processes. To do so, the approach takes as input the requirements of the user in a textual format informally describing the tasks and their ordering constraints, and generates as output a BPMN process satisfying them. This work led to a publication in an international conference [22].

6.1.4 SYNTAX Compiler Generator

Participants: Pierre Boullier, Hubert Garavel *(correspondent)*, Frédéric Lang, Wendelin Serwe.

SYNTAX is a compiler-generation tool that generates lexical analyzers (scanners) and syntactic analyzers (parsers) for deterministic and nondeterministic context-free grammars. Developed since 1970, SYNTAX is probably the oldest software of INRIA that is still actively used and maintained. In particular, SYNTAX serves to produce the front-end part of TRAIAN and of most compilers of CADP.

Since the closing of the INRIA Gforge in 2021, the SYNTAX code is hosted on the RENATER SVN repository.

In 2024, the development of SYNTAX has been particularly active, with nearly 900 commits. Significant progress was made following three directions.

Enhancements to SYNTAX core tools (“trunk”) The first part of our work focused on the core part of SYNTAX, which is used to build compiler front-ends for computer languages.

We implemented a new algorithm for recovering from syntax errors. When missing tokens have to be inserted, this algorithm gives priority to certain classes of tokens defined in advance and should thus produce repairs that are more deterministic and less dependent from small changes in the BNF grammar.

In addition of fixing two subtle bugs in the LECL tool, the “libsx” library was simplified by removing several components, and a new library “libsxbnf” was created to gather the components of the BNF tool that are used by other SYNTAX tools.

SYNTAX was ported to ARM processors running macOS and the “sxmake” script used to build SYNTAX was deeply revised to fully support multi-target compilation for several machine architectures. The C code of SYNTAX tools was cleaned by removing useless or redundant fragments, and it was adapted to the new standard C24.

The documentation was updated and a manual page for TABLE_C was written. The demo examples have been updated and equipped with test cases and scripts for executing these tests.

Enhancements to SYNTAX additional tools (“extensions” and “oldies”) Besides the core part of SYNTAX devoted to computer languages, we also worked to improve the various SYNTAX tools designed for natural-language processing, some of which are research prototypes, while the others are mature tools.

On the one hand, we screened these tools and decided to archive those that are unlikely to be used in the future. We also searched for duplicate files in order to remove them or replace them by symbolic links. We reorganized these tools by putting the code of each of them in a separate folder and we simplified the structure of these folders. We created a library “libext” that gathers the common code shared between several tools.

We revised many makefiles by bringing corrections, simplifications, and by adding “clean” targets where they were missing. The C code of these tools was also deeply revised in order to compile without

warnings using the latest versions of Gcc and Clang. We removed a lot of unused or redundant code, introduced C24 function prototypes, and fixed many problems, such as type errors and potential array overflows. This work is not complete, but significantly progressed in 2024.

Development of a FORTRAN 77 front-end with Oracle and ESOPE extensions We pursued the work undertaken in 2023 to provide the EVREF project-team (Inria Lille) with a fully functional compiler front-end for FORTRAN 77 that generates an abstract tree in JSON format.

In 2024, our F77 compiler front-end for FORTRAN 77 evolved to follow the changes brought in SYNTAX programming interfaces and makefile rules. P. Boullier extended the lexer and parser to support many extensions called by the EVREF team: DO-WHILE loops, DO loops without labels, consecutive arithmetic operators (e.g., "X * - Y"), etc. He also introduced support for other extensions proposed by Oracle (e.g., DOUBLE COMPLEX types, comment lines starting with "d", "!", etc.) or belonging to FORTRAN 90 (e.g., relational operators "==", "<=", etc.).

Based on this compiler front-end F77, we developed another front-end named ESOPE77 that addresses so-called ESOPE extensions designed for FORTRAN by CEA and used in major legacy industrial codes. In 2024, we delivered to EVREF nine successive versions of ESOPE77.

In order to test our compiler front-ends, we searched and gathered many realistic FORTRAN programs into a test suite totalling more than 1,160,000 lines of code. At the end of 2024, our F77 and ESOPE77 front-ends could process all these tests, and ESOPE77 successfully handled 97% of the code of EVREF's industrial partner.

6.2 Parallel and Distributed Verification

6.2.1 Automated Repair of Violated Eventually Properties in Concurrent Programs

Participants: Irman Faqrizal, Quentin Nivon, Gwen Salaün *(correspondent)*.

Model checking automatically verifies that a model, e.g., a Labelled Transition System (LTS), obtained from higher-level specification languages, satisfies a given temporal property. When the model violates the property, the model checker returns a counterexample, but this counterexample does not precisely identify the source of the bug. Moreover, manually correcting the given specification or model can be a painful and complicated task. To alleviate this task, we propose some techniques for computing patches that can correct an erroneous specification violating an eventually property. These techniques first extract from the whole behavioural model the part which does not satisfy the given property. In a second step, this erroneous part is analysed using several algorithms in order to compute the minimal number of patches in the specification so as to make it satisfy the given property. This work led to a publication in an international conference [16].

6.3 Timed, Probabilistic, and Stochastic Extensions

6.3.1 Quantitative Analysis of BPMN Processes

Participants: Quentin Nivon, Gwen Salaün *(correspondent)*, Ahang Zuo.

Business process optimisation is a strategic activity in organisations because of its potential to increase profit margins and reduce operational costs. We consider business processes described using an extension of BPMN with quantitative aspects for modelling execution times and resources associated with tasks. A process is not executed once but multiple times, and multiple concurrent executions of a process compete for using the shared resources. In this context, it is particularly difficult to ensure correctness and efficiency of the multiple executions of a process.

In 2024, we followed two different research directions for analyzing quantitatively business processes:

- We continued our work on process refactoring, a specific technique used for process optimisation, and we proposed a refactoring approach whose goal is to reduce the total execution time of a process. We considered BPMN processes enriched with time and resources, and executed multiple times. The proposed optimisation method consists in restructuring the processes by changing the position of their tasks. The goal of this restructuring is to limit the overuse of the resources and consequently reduce the execution time of the processes. To avoid generating unsuitable processes, the designer is involved in the restructuring phase, as s/he validates each restructuring step. This work led to a publication in an international conference [23].

- In the context of the MOAP project (see § 8.5.1), in collaboration with Yliès Falcone (CORSE project-team), we addressed resource allocation, which is a critical problem in business processes due to the simultaneous execution of tasks and resource sharing among them. The number of allocated resources affects both the execution cost and time of the process. In the context of runtime processes, a well-defined resource allocation strategy is essential for optimising waiting times and costs by mitigating delays and enhancing resource utilisation. We proposed a novel approach to dynamically adjust resource allocation during the execution of BPMN processes. The BPMN process is monitored in real-time, and the execution traces produced during its multiple executions are analysed. These execution traces are used to compute various properties or metrics of interest, including resource usage and average execution time. The approach then relies on predictive analytics to compute the future values of the aforementioned metrics. Based on these predicted results, strategies for the dynamic allocation of resources are defined, which anticipate changes in resource usage and thus dynamically update the number of resources in advance. This approach was fully automated using a toolchain and has been validated with multiple examples. This work led to a publication in an international conference [13]. A detailed account of the work carried out on the quantitative analysis of BPMN processes in the framework of MOAP is available in Ahang Zuo's PhD thesis [47].

6.4 Component-Based Architectures for On-the-Fly Verification

6.4.1 Probabilistic Runtime Enforcement of Executable BPMN Processes

Participants: Gwen Salaün *(correspondent)*, Ahang Zuo.

A business process is a collection of structured tasks corresponding to a service or a product. Business processes do not execute once and for all, but are executed multiple times, resulting in multiple instances. In this context, it is particularly difficult to ensure correctness and efficiency of the multiple executions of a process.

In 2024, in the framework of the MOAP project (see § 8.5.1), in collaboration with Yliès Falcone (CORSE project-team), we propose to rely on Probabilistic Model Checking (PMC) to automatically verify that multiple executions of a process respect some specific probabilistic property. This approach applies at runtime, thus the evaluation of the property is periodically verified and the corresponding results updated. However, we go beyond runtime PMC for BPMN, since we propose runtime enforcement techniques to keep executing the process while avoiding the violation of the property. To do so, our approach combines monitoring techniques, computation of probabilistic models, PMC, and runtime enforcement techniques. The approach has been implemented as a toolchain and has been validated on several realistic BPMN processes. This work led to a publication in an international conference [14].

6.4.2 Other Component Developments

Participants: Hubert Garavel, Frédéric Lang, Radu Mateescu, Abdelilah Mejdoubi, Wendelin Serwe.

In 2024, in addition to various bug fixes in BCG_STEADY and BCG_TRANSIENT (generation of throughput files), in XTL (compiler and "radius.xtl" library), in FSP2LOTOS and CAESAR.INDENT, in EXP.OPEN (the "-network" option), and in SVL (expansion of "leaf" and "root leaf" reduction operators), various enhancements have been brought to the following CADP tools:

- The "-diag" option of BCG_CMP now invokes BISIMULATOR to produce much smaller diagnostics, whose generation is controlled by two new options "-bfs" and "-dfs".
- BCG_STEADY and BCG_TRANSIENT were enhanced to check that, in their "parameter=value" options, all "parameter" names are pairwise distinct, and to produce throughput files with columns sorted alphabetically, easier to exploit. CUNCTATOR received two new options "-thr" and "-append", which produce an output compatible with BCG_STEADY.
- The BCG MONITOR graphical interface was deeply revised, by making its window more compact, adding two new menus "File" and "View", extending the "Edit" menu with font zooming features, adding keyboard shortcuts, and saving user settings automatically for reuse at future executions.
- The XTL language was extended with "replace" functions for the predefined type "action" and for any external XTL type defined in C, which are useful in "for" loops to update the values of accumulator variables. The predefined libraries of MCL and XTL were moved into separate directories, and the EVALUATOR 3 model checker was progressively replaced by EVALUATOR 4.
- A new option "-depth N" was added to the GENERATOR tool so as to generate an LTS fragment containing all states at distance N from the initial state.
- The SVL language was enhanced to support "condensed" priority behaviours, to control the generation of diagnostics by passing "bfs" or "dfs" options to BCG_CMP, to handle the smart reduction heuristic more efficiently, to emit clearer error messages for missing file extensions, and to make SVL scripts more concise by providing a new predefined function SVL_ECHO.
- Two new demo examples were added, namely demo 10 (collection of parallel algorithms for mutual exclusion on shared-memory machines) and demo 15 (Erlangen mainframe). The demo 40 example (Web services for stock management and on-line book auction) was translated to LNT, and several other demos were enhanced by simplifying their SVL scripts, complying with the recent checks of TRAIAN, and enriching them with related publications.

The evolutions of operating systems and C compilers dictated many changes in the CADP infrastructure, among which:

- The "mac64" version of CADP was ported to macOS 15 "Sequoia" and Xcode version 16.1. The "win64" version of CADP for Windows/Cygwin was subject to minor corrections, and the "x64" version of CADP for Windows/WSL2 (which had stopped working due to changes in WSL2 and/or the various Linux applications of the Microsoft Store) was repaired.
- Style files for Visual Studio Code have been added for various languages and file formats supported by CADP. The style files for the Sublime Text editor and EMACS have been enhanced and updated to reflect the latest changes of the LNT, MCL, and SVL languages.
- A long-term work was undertaken to upgrade the source code of CADP so that it compiles without warning using C24, the latest revision of the C language. As of April 2024, 15% of the CADP programs and code libraries had been upgraded.

6.5 Real-Life Applications and Case Studies

Formal methods are widely applied in industry, in activities ranging from the elicitation of requirements and the early design phases all the way to the deployment, configuration, and runtime monitoring of actual systems. In 2024, we participated in an international effort to survey recent successful applications of formal methods in various areas of industry [9]. The survey also brings testimonies from industry representatives, spread over Europe, Asia, North and South America, who have used formal methods in their industrial projects, covering a large spectrum of applications, not limited to the safety-critical domain.

6.5.1 Autonomous Car

Participants: Wei Chen, Jean-Baptiste Horel, Radu Mateescu *(correspondent)*, Wendelin Serwe, Aline Uwimbabazi.

Devising scenarios for testing autonomous vehicles (AV) is still a challenging task that requires a tradeoff between cost and achieved coverage of the considered operational design domain. Often scenarios are derived from accident statistics and real traffic datasets. Expertise knowledge for the tasks of selecting the scenarios for testing and/or simulation is highly required. This task is carried out mostly manually, and would benefit from a precise method to assess scenario relevance and compare scenarios.

In the framework of the PRISSMA project (see § 8.4.2) and in collaboration with Lina Marsso (University of Toronto, Canada), Christian Laugier, and Alessandro Renzaglia (CHROMA project-team), we proposed an automatic approach to generate a large number of relevant critical scenarios for autonomous driving simulators. The approach relies on the generation of behavioral conformance tests using the TESTOR tool [43], from an LNT model (specifying the ground truth configuration with the range of vehicle behaviors) and a test purpose (specifying the critical feature under analysis). The obtained test cases (which cover all possible executions exercising a given feature) are automatically translated into the inputs of autonomous driving simulators.

In 2024, in the framework of the A-IQ Ready project (see § 8.3.1), we continued this line of work along two directions:

- In collaboration with Anne Spalanzani (CHROMA project-team), we undertook the formal modeling of an AV in its environment to assist test scenario assessment and selection. We started the development of an LNT model of the interactions between an AV and groups of pedestrians in shared spaces, following the agent-based approach proposed in [45]. The pedestrian behavior is defined by several parameters capturing the physical, perceptive, and decision-making characteristics. We considered three key contexts: individual pedestrians, social groups, and interactions with AVs. This LNT model will serve as basis for generating various interaction scenarios using CADP to ensure a comprehensive coverage of pedestrian-AV interactions.

- We also used probabilistic model checking to assess the driving scenarios relevance and determine the critical ones. We considered several scenarios inspired by recent French traffic accident statistics studies, from which we selected five major scenarios, three of which involving an interaction with vehicles, and the other two involving an interaction with a pedestrian (visible and occluded). We formally modeled in LNT each driving scenario, comprising the road scenery, the behaviour of dynamic obstacles (other vehicles, pedestrians, etc.), and constraints on the behaviour of the AV (e.g., the traversal of the scene). We computed the probabilities of user-defined event sequences (e.g., a successful arrival, a collision with an obstacle, duration, perception, near misses, etc.) capturing a variety of situations. This provides a quantitative way of comparing driving scenarios, assessing their criticality, and suggesting how to improve their relevance for testing AVs.

6.5.2 Job-Shop Scheduling

Participants: Wendelin Serwe *(correspondent)*, Aline Uwimbabazi.

The job-shop scheduling problem is a well-known NP-hard scheduling problem, where a set of jobs (consisting of a sequence of tasks) have to be executed on a set of machines, each task specifying its duration and the required machine.

In 2024, in the framework of the A-IQ Ready project (see § 8.3.1), we considered the Job-Shop Scheduling Problem to compute schedules for autonomous robots. We aimed at studying whether the state-space exploration algorithms of CADP can be used to compute *optimal* schedules. We have experimented several encodings of the problem in the LNT language.

Our models are based on the heuristics approach to execute the tasks as soon as possible, i.e., if a machine m is free and there is some job searching to execute a task t on m, then t is immediately assigned to m. The main objective of our research work was to develop efficient models that can perform better by providing better solutions (than those found in some benchmark instances and the ones found by using reinforcement learning), where the execution of all jobs can be completed as fast as possible, while still considering assumptions such as, respecting the order of tasks for execution in each job, assigning each operation to one and only one of its eligible machines (no interruption of an operation is allowed once it is started), and allowing each machine to perform at most one task at a time. Our models can be used to study the Job-Shop Scheduling Problem because we were able to generate the complete Labelled Transition System (LTS) of a given instance, which allowed us to find all possible solutions.

6.5.3 IEEE 1394 Link Layer

Participants: Hubert Garavel.

IEEE 1394 (also called "FireWire") is an interface standard that specifies a serial bus architecture for high-speed communications. It can connect up to 63 peripherals in a tree or daisy-chain topology, and can perform both asynchronous and isochronous transfers simultaneously. It was developed between 1987 and 1995 by a large consortium gathering Apple, Panasonic, Philips, Sony, and many others contributors. This work resulted in an IEEE standard, followed by integration in many industrial products (in particular, Apple products until 2016).

In the framework of the COST-247 action, a pan-European academic collaboration that took place between 1994 and 1997, the asynchronous mode of the link layer protocol of IEEE 1394 was selected as an interesting case study for formal methods. At CWI Amsterdam, Bas Luttik developed a formal model in the μCRL language and stated five correctness properties that the protocol should satisfy. At INRIA Grenoble, Mihaela Sighireanu translated this model to LOTOS and, using the XTL model checker with the help of R. Mateescu, discovered that the deadlock-freeness property did not hold, i.e., that the protocol could enter a deadlock state after a specific sequence of 50 transitions [46].

In 2024, in collaboration with Bas Luttik (Eindhoven University of Technology, The Netherlands), we revisited this case study to present, along with the original μCRL model, three companion models: a model written in mCRL2 by Jan Friso Groote, a recent revision of the LOTOS model developed by M. Sighireanu, and a novel model written in LNT, which is now available as the CADP demo 23. This work led to a publication in an international workshop [18].

6.5.4 Erlangen Mainframe

Participants: Hubert Garavel.

The Erlangen mainframe was a multiprocessor computer for processing jobs of different priorities and subject to hardware failures. The formal modelling and quantitative analysis of the Erlangen mainframe was proposed in 1994 as a challenging problem by Ulrich Herzog and Vassilis Merksiotakis (IMMD-7 team, Erlangen University).

Using the stochastic process algebra TIPP, a formal model of this mainframe was specified, which makes intensive use of parallel composition, multiway synchronisation between two or more concurrent processes, and compound transitions combining synchronised actions with rates of Continuous-Time Markov Chains. From this formal model, probabilistic results about availability, performability, and proper dimensioning of the mainframe were obtained [39, 38] using the TIPP software tools, which are no longer maintained.

In 2024, in collaboration with Holger Hermanns (Saarland University, Germany) and Dave Parker (University of Oxford, United Kingdom), we developed two novel formal models for the Erlangen mainframe, one in the automata-based language PRISM, and another one in LNT. Using three different tools, namely

PRISM, Storm, and CADP, we performed all numerical experiments and successfully reproduced the same figures as in the original publications. The fact that three different tools developed independently give identical results on the same model suggest that these tools could be used in combination to support safety/security certification of critical systems. This work led to an international publication [24].

6.5.5 Manufacturing Application

Participants: Irman Faqrizal, Gwen Salaün *(correspondent)*.

The ever-increasing complexity of industrial automation systems generates a demand for reliable development methods. IEC 61499 is a recent standard devoted to the development of industrial automation systems by promoting reusability, reconfigurability, interoperability, and portability.

In 2024, in the framework of the D-IIoT project (see § 8.5.2), we contributed to the formal modeling and analysis of IEC 61499 automation systems along two directions:

- Formal verification techniques, such as model checking, are useful to ensure the correctness of IEC 61499 applications at design time. However, they do not consider the impact of the environment on the application behaviour at runtime. In collaboration with Tatiana Liakh, Midhun Xavier, and Valeriy Vyatkin (Lulea University of Technology, Sweden), we combined design time and runtime analyses to apply probabilistic model checking on an IEC-61499-based manufacturing application. We proposed several probabilistic properties to be checked, and visualise the results graphically for analysis purposes, making possible to optimise the system's quantitative features, such as productivity. This work led to a publication in an international conference [15].

- During their life cycle, industrial automation systems need to evolve according to requirements, and modifying the applications to satisfy these requirements can be complex and error-prone. In collaboration with Yliès Falcone (CORSE project-team), we proposed techniques to guide the evolution of IEC 61499 applications. Given an initial application and the evolution requirements, we generate guidelines for modifying the application to satisfy the requirements. The application is first translated into a behavioural model describing all possible sequences of events the application can trigger. We then apply algorithms to extract relevant submodels of the application and modify them according to the requirements. Finally, the submodels are analysed to generate guidelines for modifying the application. These guidelines can bridge the gap between the requirements and the target application. Instead of only considering the requirements when exploring possible modifications, the developers can use the guidelines to make necessary changes to the application. A mixing tank system is used as a running example to illustrate the approach. In addition, a prototype to automate the evolution techniques is developed. This work led to a publication in an international conference [17] and a publication in an international journal [11].

A detailed account of the work carried out on industrial automation systems in the framework of D-IIoT is available in Irman Faqrizal's PhD thesis [29].

6.5.6 Systems-on-Chip

Participants: Philippe Ledent, Radu Mateescu *(correspondent)*, Wendelin Serwe.

SoC (System-on-Chip) architectures, which are widely used in smartphones and IoT devices, integrate a complete computing system on a single chip. The complexity of SoCs, together with their potential use in critical applications, call for thorough verification; however, the current industrial practice is still mainly based on hardware simulators using directed tests and/or execution-time assertions. A major challenge in SoC verification is to devise meaningful system-level tests involving many components of the SoC.

PSS (*Portable test and Stimulus Standard*), defined by the Accellera consortium of hardware design tool vendors and users, provides a language to specify both a verification intent (e.g., a sequence of actions) and an implicit description of the SoC's behavior (as a collection of ordering constraints on the actions). PSS also defines a methodology to derive from a verification intent, by automatic inference of (intermediate) actions, complete tests to be executed on the actual SoC.

In 2024, we sought to improve the testing methodologies for SoCs along two directions:

- Since PSS emphasizes the verification intent and limits the modeling of the SoC's behavior to a set of constraints, the actually modeled behavior becomes difficult to grasp. Also, PSS promotes the derivation of complete tests using a backward exploration of the verification intent that favors shortest-path solutions, possibly missing interesting and more complex tests. To improve the PSS testing methodology, we suggested to formally express the behavior of a PSS model as a composition of communicating LTSs, obtained by an automated translation from PSS to the LNT and EXP languages of CADP. This makes possible to formally verify temporal logic properties of the model and thus increase the confidence in the model and the generated tests. Also, applying conformance testing techniques with coverage guarantees [42] enables to improve the coverage of the generated tests. This work led to a publication in an international conference [20].

- Hardware attacks consist in forcing a SoC to perform operations in order to access functionalities or information that should normally not be available. A critical security requirement is resource isolation, which forbids applications (or programs) running on a same SoC to access data not intended for them. Even though there is still no generally accepted technique to generate appropriate tests, it became clear that tests should be generated at the system level. We illustrated the modeling aspects in test generation for resource isolation, namely modeling the behavior and expressing the intended test scenario. We studied both aspects using the industrial standard PSS and an academic approach based on conformance testing. This work led to a publication in an international conference [21].

A detailed account of the work carried out on formal modeling and analysis of SoCs is available in Ph. Ledent's PhD thesis [40].

7 Bilateral contracts and grants with industry

7.1 Bilateral grants with industry

7.1.1 Euris

Participants: Suraj Gupta, Frédéric Lang, Gwen Salaün *(correspondent)*.

S. Gupta is supported by a CIFRE grant (from February 2024 to January 2027) from Euris (Paris) on the formal modeling and analysis of blockchain protocols in the health domain, under the supervision of Gwen Salaün, Frédéric Lang, and Umar Ozeer (Euris).

8 Partnerships and cooperations

8.1 International initiatives

H. Garavel is a member of IFIP (*International Federation for Information Processing*) Technical Committee 1 (*Foundations of Computer Science*) Working Group 1.8 on Concurrency Theory chaired successively by Luca Aceto and Jos Baeten.

8.2 International research visitors

8.2.1 Visits of international scientists

- Holger Hermanns (Saarland University, Germany) visited our team on October 17–18, 2024.

8.2.2 Visits to international teams

Research stays abroad

- H. Garavel visited Saarland University (Germany) on November 8–12, 2024.

8.2.3 Other international collaborations

In 2024, we had scientific relations with several universities and institutes abroad, including:

- Technical University of Eindhoven, The Netherlands (Bas Luttik, Jan Friso Groote),
- University of Málaga, Spain (Francisco Durán),
- University of Oxford, United Kingdom (Dave Parker),
- Saarland University, Germany (Holger Hermanns),
- University of Toronto, Canada (Lina Marsso).

8.3 European initiatives

8.3.1 Horizon Europe

A-IQ Ready

Participants: Frédéric Lang, Radu Mateescu *(correspondent)*, Wendelin Serwe.

A-IQ Ready project on cordis.europa.eu

Title: Artificial Intelligence using Quantum measured Information for realtime distributed systems at the edge

Duration: From January 1, 2023 to December 31, 2025

Partners:

- SCALIRO GMBH, Germany
- BUNDESMINISTERIUM FUER LANDESVERTEIDIGUNG (BMLV), Austria
- UAB TERAGLOBUS, Lithuania
- INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE (INRIA), France
- SAFELOG GMBH, Germany
- AVL ARASTIRMA VE MUHENDISLIK SANAYI VE TICARET LIMITED SIRKETI (AVL TURKIYE), Türkiye
- EMOTION3D GMBH, Austria
- AIT AUSTRIAN INSTITUTE OF TECHNOLOGY GMBH (AIT), Austria
- SYNOPSYS NETHERLANDS BV (VIRAGE LOGIC), Netherlands
- TECHNISCHE UNIVERSITAET GRAZ (TU GRAZ), Austria
- IDEAS & MOTION SRL, Italy

- CHAROKOPEIO PANEPISTIMIO (HAROKOPIO UNIVERSITY OF ATHENS (HUA)), Greece
- MONTANUNIVERSITAET LEOBEN (Montanuniversitaet Leoben), Austria
- TEKNE SRL (TEKNE), Italy
- IL-INGENIEURBURO LAABMAYR & PARTNER ZT GESMBH (Laabmayr), Austria
- SLEEP ADVICE TECHNOLOGIES SRL, Italy
- HUAWEI TECHNOLOGIES SWEDEN AB (HWSE), Sweden
- INFORMATION TECHNOLOGY FOR MARKET LEADERSHIP (ITML), Greece
- UNIKIE OY (UNIKIE), Finland
- MERCEDES-BENZ AG, Germany
- IOBUNDLE, LDA, Portugal
- TEKNOLOGIAN TUTKIMUSKESKUS VTT OY (VTT), Finland
- KOUVOLA INNOVATION OY, Finland
- TECHNISCHE UNIVERSITAET MUENCHEN (TUM), Germany
- INESC TEC - INSTITUTO DE ENGENHARIADE SISTEMAS E COMPUTADORES, TECNOLOGIA E CIENCIA (INESC TEC), Portugal
- INSTITUT MIKROELEKTRONICKYCH APLIKACI SRO (IMA), Czechia
- HOCHSCHULE OFFENBURG, Germany
- TECHNISCHE HOCHSCHULE ROSENHEIM / TECHNICAL UNIVERSITY OF APPLIED SCIENCES (TECHNISCHE HOCHSCHULE ROSENHEIM), Germany
- N VISION SYSTEMS AND TECHNOLOGIES SL (NVISION), Spain
- VAISTO SOLUTIONS OY, Finland
- INSAR.SK SRO (INSAR.SK), Slovakia
- VYSOKE UCENI TECHNICKE V BRNE (BRNO UNIVERSITY OF TECHNOLOGY), Czechia
- OSTBAYERISCHE TECHNISCHE HOCHSCHULEAMBERG-WEIDEN (OTH Amberg-Weiden), Germany
- MANTSINEN GROUO LTD OY, Finland
- VIRTUAL VEHICLE RESEARCH GMBH (VIF), Austria
- METSA FIBRE OY (POHJAN SELLU KEMI BOTNIA PULPS METSA-RAUMA METSA-BOTNIA), Finland
- INNATERA NANOSYSTEMS BV, Netherlands
- PUMACY TECHNOLOGIES AG (PUMACY), Germany
- CARGOTEC FINLAND OY (KALMAR), Finland
- UNIVERSIDAD DE ALCALA (UNIVERSIDAD DE ALCALA), Spain
- SILICON MOBILITY (SILICON MOBILITY), France
- POLITECNICO DI TORINO (POLITO), Italy
- AVL LIST GMBH (AVL), Austria
- TTTECH AUTO AG, Austria
- TTTECH COMPUTERTECHNIK AG, Austria
- ELEKTRONIKAS UN DATORZINATNU INSTITUTS (EDI), Latvia
- UNIVERSITAET zu LUEBECK (UZL), Germany
- UNIVERSITA DEGLI STUDI DI MODENA E REGGIO EMILIA (UNIMORE), Italy
- UNIVERSIDAD POLITECNICA DE MADRID (UPM), Spain
- ARQUIMEA RESEARCH CENTER SL, Spain

Inria contact: Radu Mateescu

Coordinator: Katrin Al Jezany (AVL)

Summary: Global environmental issues, social inequality and geopolitical changes will pose numerous problems for our society in the future. To face these new challenges and deal with them, there is a need to understand and appropriately utilize new digital technologies such as artificial intelligence (AI), the Internet of Things (IoT), robotics and biotechnologies.

A-IQ Ready proposes cutting-edge quantum sensing, edge continuum orchestration of AI and distributed collaborative intelligence technologies to implement the vision of intelligent and autonomous ECS for the digital age. Quantum magnetic flux and gyro sensors enable highest sensitivity and accuracy without any need for calibration, offer unmatched properties when used in combination with a magnetic field map. Such a localization system will enhance the timing and accuracy of the autonomous agents and will reduce false alarms or misinformation by means of AI and multi-agent system concepts. As a priority, the communication guidance and decision making of groups of agents need to be based on cutting-edge technologies. Edge continuum orchestration of AI will allow decentralizing the development of applications, while ensuring an optimal use of the available resources. Combined with the quantum sensors, the edge continuum will be equipped with innovative, multi-physical capabilities to sense the environment, generating "slim" but accurate measurements. Distributed intelligence will enable emergent behavior and massive collaboration of multiple agents towards a common goal. By exploring the synergies of these cutting-edge technologies through civil safety and security, digital health, smart logistics for supply chains and propulsion use cases, A-IQ Ready will provide the basis for the digital society in Europe based on values, moving towards the ideal of Society 5.0.

The main contributions of CONVECS are the formal modeling and validation of intelligent transportation systems and indoor logistics applications.

8.3.2 Other european programs/initiatives

The CONVECS project-team is member of the FMICS (Formal Methods for Industrial Critical Systems) working group of ERCIM. H. Garavel and R. Mateescu are members of the FMICS board, H. Garavel being in charge of dissemination actions.

8.4 National initiatives

8.4.1 PEPR Cloud

The cloud has become an essential ingredient of IT systems. Its model, initially based on large data centers, has evolved towards "edge computing" or "digital continuum", developing secure, interoperable hybrid clouds to process large volumes of data quickly and efficiently. This change is crucial for future applications such as smart cities or autonomous vehicles. However, cloud software needs to be rethought to adapt to the diversity of operators, multiple access points, user and resource mobility, while addressing the challenges of security and energy consumption control.

The PEPR Cloud is a research program promoted by the French government as part of the France 2030 "Cloud" strategy to develop secure and frugal Cloud technologies. PEPR Cloud aims to advance Cloud technologies and facilitate the transfer of innovations and solutions from research to industry. PEPR Cloud started in April 2024 for seven years. CONVECS is involved in two projects of PEPR Cloud, described below.

Archi-CESAM

Participants: Zachary Assoumani, Hubert Garavel, Radu Mateescu *(correspondent)*, Wendelin Serwe.

Archi-CESAM (*Converged, Efficient and Safe Architecture based on Near Memory Accelerators*) is a PEPR Cloud project led by CEA and involving five other partners (Inria, Université de Rennes, Télécom SudParis, Grenoble INP, and CNRS). The project proposes to rethink hardware (computation, memory and interconnection) so that it is co-designed with the application in a converged and trusted architecture perspective, in an environment known for its abundance of data to be processed. Archi-CESAM tackles the major cloud evolution (increased parallelism, specialization, new interconnections, virtualization) with a global, coordinated approach to distributed architectures, acceleration, interconnection and security bricks, not forgetting design methods.

The main contribution of CONVECS to Archi-CESAM is a rigorous methodology of designing hardware architectures, based on formal methods, verification, and conformance test generation.

TARANIS

Participants: Ahmed Khebbeb, Gwen Salaün *(correspondent)*.

TARANIS (*Model, Deploy, Orchestrate, and Optimize Cloud Applications and Infrastructure*) is a PEPR Cloud project led by Inria Lyon and involving nine other partners (CNRS, IMT, UGA, CEA, Université de Rennes, ENS Lyon, Université Claude Bernard, Université de Lille, INSA Rennes). The project proposes to exploit the new cloud infrastructures (edge computing, digital continuum) efficiently by abstracting the description of applications and resources to automate their management even further. This will enable a global optimization of resources according to multi-criteria objectives (price, deadline, performance, energy, etc.) on both the user side (applications) and the resource provider side (infrastructures). TARANIS also addresses the challenges of abstracting application reconfiguration and dynamically adapting resource usage.

The main contribution of CONVECS to TARANIS is on resource provisioning and orchestration languages and platforms that are used for deploying and updating cloud applications. More precisely, we plan to develop formal specifications and models as well as analysis capabilities for verifying functional and non-functional properties on such models.

8.4.2 Fonds pour l'innovation et l'industrie

PRISSMA

Participants: Jean-Baptiste Horel, Radu Mateescu *(correspondent)*.

PRISSMA (*Plateforme de Recherche et d'Investissement pour la Sûreté et la Sécurité de la Mobilité Autonome*) is a project funded by the *Fonds pour l'innovation et l'industrie* within the *Grand défi 2 : sécuriser, certifier et fiabiliser les systèmes fondés sur l'intelligence artificielle* programme. The project involves 19 industrial partners (among which ANSYS, RATP, and VALEO), as well as Université Gustave-Eiffel, LNE, and Inria (project-teams CHROMA and CONVECS). PRISSMA aims at proposing a platform enabling to release the technological locks that hamper the deployment of secure IA-based systems and to integrate all the necessary elements for the homologation activities of autonomous vehicles and their validation in real environments given by use cases.

PRISSMA started in April 2021 for three years. The main contributions of CONVECS to PRISSMA are the formal modeling and validation of perception components of the autonomous vehicle.

8.4.3 Other national collaborations

We had sustained scientific relations with the following researchers:

- César Fuguet (MADMAX project-team),
- Mario Cortes Cornax, Akram Idani, Lucie Muller, and Germán Vega (VASCO team, LIG),
- Nicolas Anquetil, Stéphane Ducasse, and Larisa Safina (EVREF project-team, Lille).

8.5 Regional initiatives

8.5.1 Region Auvergne-Rhône-Alpes

Participants: Gwen Salaün *(correspondent)*, Ahang Zuo.

MOAP is a project funded by the Auvergne-Rhône-Alpes region within the *Pack Ambition Recherche* programme. The project involves the project-teams CONVECS and CORSE, and the SOITEC company. MOAP aims at providing modelling and automated analysis techniques for enabling companies to master the complexity of their internal processes and for optimizing those processes with the final goal of improving the quality and productivity of their businesses.

MOAP started in October 2020 for five years. The main contributions of CONVECS to MOAP are the formal modeling and automated verification of BPMN processes.

8.5.2 Persyval Labex

Participants: Irman Faqrizal, Gwen Salaün *(correspondent)*.

D-IIoT is a project funded by the Persyval Labex via the ANR ("*Agence Nationale de la Recherche*"). The project involves research teams of three local laboratories (CEA, LIG, VERIMAG). D-IIoT aims at studying and proposing new techniques to support the execution of long-running and evolving IIoT (Industrial IoT) applications with dependability guarantees (e.g., security, correctness).

D-IIoT started in October 2021 for three years. The main contributions of CONVECS to D-IIoT are the formal modeling, verification and reconfiguration of IIoT applications.

9 Dissemination

Participants: Hubert Garavel, Frédéric Lang, Philippe Ledent, Radu Mateescu, Quentin Nivon, Gwen Salaün, Wendelin Serwe.

9.1 Promoting scientific activities

9.1.1 Scientific events: organisation

General chair, scientific chair

- Together with Peter Höfner (Data61, CSIRO, Sydney, Australia), H. Garavel set up a model repository to collect and archive formal models of real systems; this infrastructure is used by the series of MARS workshops. This repository currently contains 37 models, among which 11 were deposited by CONVECS.
- H. Garavel is a member of the model board of MCC (*Model Checking Contest*).
- H. Garavel is a member of the steering committee of the MARS (*Models for Formal Analysis of Real Systems*) workshop series since 2015.
- H. Garavel is a member of the steering committee of TTC (*Transformation Tool Contest*) since 2021.
- H. Garavel and R. Mateescu are members of the steering committee of the FMICS (*Formal Methods for Industrial Critical Systems*) conference series since 2018.
- G. Salaün is member of the steering committee of the FACS (*Formal Aspects of Component Software Symposium*) conference series since 2021.

- G. Salaün is member of the steering committee of the ACM SAC-SVT (*Symposium of Applied Computing – Software Verification and Testing track*) conference series since 2018.
- G. Salaün is member of the steering committee of the SEFM (*International Conference on Software Engineering and Formal Methods*) conference series since 2014.

9.1.2 Scientific events: selection

Chair of conference program committees

- F. Lang was programme committee co-chair of MARS'2024 (*6th International Workshop on Models for Formal Analysis of Real Systems*), Luxembourg City, Luxembourg, April 6, 2024.
- W. Serwe was programme committee co-chair of FMICS'2024 (*29th International Conference on Formal Methods for Industrial Critical Systems*), Milan, Italy, September 9-11, 2024.

Member of the conference program committees

- R. Mateescu was a programme committee member of SPIN'2024 (*30th International Symposium on Model Checking Software*), Luxembourg City, Luxembourg, April 10-11, 2024.
- R. Mateescu was a programme committee member of IFIP-ICTSS'2024 (*36th IFIP International Conference on Testing Software and Systems*), London, UK, October 30-November 1st, 2024.
- G. Salaün was a programme committee member of SAC/SVT'2024 (*39th ACM/SIGAPP Symposium on Applied Computing - Software Verification and Testing Track*), Avila, Spain, April 8-12, 2024.
- G. Salaün was a programme committee member of ENASE'2024 (*19th International Conference on Evaluation of Novel Approaches to Software Engineering*), Angers, France, April 28-29, 2024.
- G. Salaün was a programme committee member of COMPSAC-SETA'2024 (*IEEE International Conference on Computers, Software, and Applications - Software Engineering Technologies and Applications*), Osaka, Japan, July 2-4, 2024.
- G. Salaün was a programme committee member of ICSOFT'2024 (*19th International Conference on Software Technologies*), Dijon, France, July 8-10, 2024.
- G. Salaün was a programme committee member of FM-BPM'2024 (*2nd International Workshop on Formal Methods for Business Process Management*), Krakow, Poland, September 2, 2024.
- G. Salaün was a programme committee member of FACS'2024 (*20th International Conference on Formal Aspects of Component Software*), Milan, Italy, September 9-10, 2024.
- G. Salaün was a programme committee member of DATAMOD'2024 (*12th International Symposium "From Data to Models and Back"*), Aveiro, Portugal, November 4-5, 2024.
- G. Salaün was a programme committee member of SEFM'2024 (*22nd International Conference on Software Engineering and Formal Methods*), Aveiro, Portugal, November 4-8, 2024.
- W. Serwe was a programme committee member of ASYDE'2024 (*6th International Workshop on Automated and verifiable Software sYstem DEvelopment*), Sacramento, California, USA, October 28, 2024.

Reviewer

- Q. Nivon was a reviewer for COMPSAC-SETA'2024, DATAMOD'2024, ENASE'2024, SAC/SVT'2024, SEFM'2024, and an artifact reviewer for SPIN'2024.
- W. Serwe was a reviewer for DATE'2025 (*Design, Automation and Test in Europe*), IFIP-ICTSS'2024, SEFM'2024, and SPIN'2024.

9.1.3 Journal

Member of the editorial boards

- H. Garavel is a member of the editorial board of STTT (*Springer International Journal on Software Tools for Technology Transfer*).

Reviewer - reviewing activities

- F. Lang was a reviewer for ESWA (*Expert Systems With Applications*) and STTT.
- R. Mateescu was a reviewer for TR (*IEEE Transactions on Reliability*) and SCP (*Science of Computer Programming*).
- G.Salaün was a reviewer for Information Systems, ISA Transactions, JSS (*Journal of Systems and Software*), Supercomputing, and TAAS (*ACM Transactions on Autonomous and Adaptive Systems*).
- W. Serwe was a reviewer for SCP.

9.1.4 Software Dissemination and Internet Visibility

The CONVECS project-team distributes several software tools, including the CADP toolbox.

In 2024, the main facts are the following:

- We prepared and distributed twelve successive versions (2024-a to 2024-l) of CADP.
- We granted CADP licenses for 378 different computers in the world.

The CONVECS Web site was updated with scientific contents, announcements, publications, etc.

By the end of December 2024, the CADP forum, opened in 2007 for discussions regarding the CADP toolbox, had over 482 registered users and over 1987 messages had been exchanged.

Also, for the 2024 edition of the Model Checking Contest, we provided 4 families of models (totalling 84 Nested-Unit Petri Nets) derived from our LNT models.

9.1.5 Invited talks

- Ph. Ledent gave a talk entitled "*Enhancing SoC Architecture Verification*" at the "Café Tech Pizza" event organized at Grenoble INP on September 19, 2024.
- Ph. Ledent and Q. Nivon participated to the workshop organized by the MFML (*Méthodes Formelles, Modèles et Langages*) axis of the LIG laboratory (Grenoble) on May 29, 2024. Ph. Ledent gave a talk entitled "*Testing Resource Isolation for System-on-Chip Architectures*". Q. Nivon gave a talk entitled "*From Textual Requirements to BPMN*".
- G. Salaün gave a talk entitled "*Automated Verification of Textual Process Descriptions*" at the Colloquium of the Formal System Analysis research group at the Eindhoven University of Technology (The Netherlands) on November 5, 2024.

9.1.6 Research administration

- R. Mateescu is the scientific correspondent of the International Partnerships for Inria Grenoble.
- R. Mateescu is a member of the "*Comité d'Orientation Scientifique*" for Inria Grenoble.
- R. Mateescu is representative of Inria Grenoble at the International Relations and Outreach of Université Grenoble Alpes (UGA).
- R. Mateescu was a member of the council of the Mathematics, Information and Communication Sciences (MSTIC) research department of UGA until November 2024.
- G. Salaün is the director of the MSTIC research department of UGA.

- G. Salaün was the head of the *Métiers du Multimédia et de l'Internet* (MMI) department at IUT1/UGA until August 31, 2024.
- W. Serwe is the chair of the "*Commission du développement technologique*", which is in charge of selecting R&D projects for Inria Grenoble, and giving an advice on the recruitment of temporary engineers.
- W. Serwe is a member of the "*Comité de Centre*" at Inria Grenoble.

9.2 Teaching - Supervision - Juries

9.2.1 Teaching

CONVECS is a host team for the computer science master MOSIG (*Master of Science in Informatics at Grenoble*), common to Grenoble INP and UGA.

In 2024, we carried out the following teaching activities:

- F. Lang gave a course on "*Formal Software Development Methods*" (7.5 hours "*équivalent TD*") in the framework of the "*Software Engineering*" lecture given to first year students of the MOSIG.
- F. Lang gave a course on "*Programming Languages, Compilers, and Semantics*" (27 hours "*équivalent TD*") to first year students of the MOSIG and of the Master 1 Informatique.
- F. Lang and R. Mateescu gave a lecture on "*Modeling and Analysis of Concurrent Systems: Models and Languages for Model Checking*" (27 hours "*équivalent TD*") to third year students of ENSIMAG.
- F. Lang gave a course on "*Modeling and Analysis of Asynchronous Concurrent Systems*" (15 hours "*équivalent TD*") in the framework of the "*Safety Critical Systems*" lecture given to second year students of the MOSIG.
- Q. Nivon gave a course on "*Gestion de données relationnelles et applications*" (39 hours "*équivalent TD*") to L2 students of UGA.
- Q. Nivon gave a course on "*Sémantique des langages de programmation et compilation*" (30 hours "*équivalent TD*") to M1 students of UGA.
- G. Salaün taught about 200 hours of classes (algorithmics, Web development, object-oriented programming) at the MMI department of IUT1/UGA.
- W. Serwe supervised a group of six teams in the context of the "*projet Génie Logiciel*" (55 hours "*équivalent TD*", consisting in 13.5 hours of lectures, plus supervision and evaluation), ENSIMAG, January 2024.

9.2.2 Supervision

- PhD: I. Faqrizal, "*Quantitative Verification and Runtime Techniques for Industrial Automation Systems*", Université Grenoble Alpes, defended on December 5, 2024, G. Salaün and Y. Falcone
- PhD: P. Ledent, "*Formal Modeling for Testing of System-on-Chip Resource Isolation*", Université Grenoble Alpes, defended on October 24, 2024, R. Mateescu, W. Serwe, and Hajer Ferjani (ST Microelectronics)
- PhD: Chukri Soueidi, "*Ingénierie de l'instrumentation pour la vérification de l'exécution*", Université Grenoble Alpes, defended on May 13, 2024, G. Salaün and Y. Falcone
- PhD: A. Zuo, "*Modelling, Runtime Analysis and Quantitative Verification of Business Processes*", Université Grenoble Alpes, defended on April 11, 2024, G. Salaün and Y. Falcone
- PhD in progress: Z. Assoumani, "*Conception rigoureuse de circuits basée sur les méthodes formelles*", Université Grenoble Alpes, since October 2024, R. Mateescu and W. Serwe

- PhD in progress: S. Gupta, "*Using blockchains for managing EHRs (Electronic Health Records)*", Université Grenoble Alpes, since January 2024, G. Salaün, F. Lang, and Umar Ozeer (Euris)
- PhD in progress: J-B. Horel, "*Validation des composants de perception basés sur l'IA dans les véhicules autonomes*", Université Grenoble Alpes, since April 2021, R. Mateescu, Alessandro Renzaglia, and Christian Laugier (CHROMA project-team)
- PhD in progress: A. Khebbeb, "*Formal Modelling and Automated Analysis of Resource Provisioning Languages*", Université Grenoble Alpes, since December 2024, G. Salaün and Philippe Merle (SPIRALS project-team, Lille)
- PhD in progress: Q. Nivon, "*Analyse, optimisation et debugging de processus BPMN*", Université Grenoble Alpes, since October 2022, G. Salaün

9.2.3 Juries

- G. Salaün was reviewer of Olav Bunte's PhD thesis, entitled "*Cracking OIL: A Formal Perspective on an Industrial DSL for Modelling Control Software*", defended at Eindhoven University of Technology (The Netherlands) on September 5, 2024.
- G. Salaün was jury president for Giovanni Fabbretti's PhD thesis, entitled "*A Theory of Distributed Reversible Systems with Failures and Recovery*", defended at UGA on October 11, 2024.
- G. Salaün was jury president for Pietro Lami's PhD thesis, entitled "*Reversibility for Concurrent Memory Models*", defended at UGA on December 16, 2024.

10 Scientific production

10.1 Major publications

[1] G. Barbon, V. Leroy and G. Salaün. 'Debugging of Behavioural Models using Counterexample Analysis'. In: *IEEE Transactions on Software Engineering* 47.6 (June 2021), pp. 1184–1197. DOI: 10.1109/TSE.2019.2915303. URL: https://inria.hal.science/hal-02145610.

[2] H. Evrard and F. Lang. 'Automatic Distributed Code Generation from Formal Models of Asynchronous Processes Interacting by Multiway Rendezvous'. In: *Journal of Logical and Algebraic Methods in Programming* 88 (Mar. 2017), p. 33. DOI: 10.1016/j.jlamp.2016.09.002. URL: https://hal.inria.fr/hal-01412911.

[3] H. Garavel. 'Nested-unit Petri nets'. In: *Journal of Logical and Algebraic Methods in Programming* 104 (Apr. 2019), pp. 60–85. DOI: 10.1016/j.jlamp.2018.11.005. URL: https://hal.inria.fr/hal-02072190 (cit. on p. 11).

[4] H. Garavel, F. Lang and R. Mateescu. 'Compositional Verification of Asynchronous Concurrent Systems using CADP'. In: *Acta Informatica* 52.4 (June 2015), p. 56. DOI: 10.1007/s00236-015-0226-1. URL: https://hal.inria.fr/hal-01247507.

[5] H. Garavel, F. Lang, R. Mateescu and W. Serwe. 'CADP 2011: A Toolbox for the Construction and Analysis of Distributed Processes'. In: *International Journal on Software Tools for Technology Transfer* 15.2 (2013), pp. 89–107. DOI: 10.1007/s10009-012-0244-z. URL: http://hal.inria.fr/hal-00715056 (cit. on p. 7).

[6] H. Garavel, F. Lang and W. Serwe. 'From LOTOS to LNT'. In: *ModelEd, TestEd, TrustEd - Essays Dedicated to Ed Brinksma on the Occasion of His 60th Birthday*. Ed. by J.-P. Katoen, R. Langerak and A. Rensink. Vol. 10500. Lecture Notes in Computer Science. Springer, Oct. 2017, pp. 3–26. DOI: 10.1007/978-3-319-68270-9_1. URL: https://hal.inria.fr/hal-01621670 (cit. on pp. 3, 9).

[7] A. Krishna, P. Poizat and G. Salaün. 'Checking Business Process Evolution'. In: *Science of Computer Programming* 170 (Jan. 2019), pp. 1–26. DOI: 10.1016/j.scico.2018.09.007. URL: https://hal.inria.fr/hal-01920273.

[8] R. Mateescu and W. Serwe. 'Model Checking and Performance Evaluation with CADP Illustrated on Shared-Memory Mutual Exclusion Protocols'. In: *Science of Computer Programming* (Feb. 2012). DOI: 10.1016/j.scico.2012.01.003. URL: http://hal.inria.fr/hal-00671321.

10.2 Publications of the year

International journals

[9] M. ter Beek, R. Chapman, R. Cleaveland, H. Garavel, R. Gu, I. ter Horst, J. Keiren, T. Lecomte, M. Leuschel, K. Y. Rozier, A. Sampaio, C. Seceleanu, M. Thomas, T. Willemse and L. Zhang. 'Formal Methods in Industry'. In: *Formal Aspects of Computing* (21st Aug. 2024), pp. 1–34. DOI: 10.1145/3689374. URL: https://inria.hal.science/hal-04776404 (cit. on p. 15).

[10] L. Di Stefano and F. Lang. 'Compositional verification of priority systems using sharp bisimulation'. In: *Formal Methods in System Design* 62 (2024), pp. 1–40. DOI: 10.1007/s10703-023-00422-1. URL: https://inria.hal.science/hal-04103681.

[11] I. Faqrizal, G. Salaün and Y. Falcone. 'Adaptive Industrial Control Systems via IEC 61499 and Runtime Enforcement'. In: *ACM Transactions on Autonomous and Adaptive Systems* (2024), pp. 1–31. DOI: 10.1145/3691345. URL: https://inria.hal.science/hal-04680168. In press (cit. on p. 18).

International peer-reviewed conferences

[12] P. Bouvier and H. Garavel. 'Identifying Duplicates in Large Collections of Petri Nets and Nested-Unit Petri Nets'. In: *Lecture Notes in Computer Science*. PETRI NETS 2024 - 45th International Conference on Application and Theory of Petri Nets and Concurrency. Vol. 14628. Genève, Switzerland: Springer Nature Switzerland, 13th June 2024, pp. 379–401. URL: https://inria.hal.science/hal-04776402 (cit. on p. 11).

[13] Y. Falcone, G. Salaün and A. Zuo. 'Dynamic Resource Allocation for Executable BPMN Processes Leveraging Predictive Analytics'. In: QRS 2024 - 24th International Conference on Software Quality, Reliability, and Security. Cambridge, United Kingdom, 2024, pp. 1–12. URL: https://inria.hal.science/hal-04617808 (cit. on p. 14).

[14] Y. Falcone, G. Salaün and A. Zuo. 'Probabilistic Runtime Enforcement of Executable BPMN Processes'. In: FASE 2024 - 27th International Conference on Fundamental Approaches to Software Engineering. Luxembourg City, Luxembourg, 8th Apr. 2024, pp. 1–21. DOI: 10.1007/978-3-031-57259-3_3. URL: https://inria.hal.science/hal-04533195 (cit. on p. 14).

[15] I. Faqrizal, T. Liakh, M. Xavier, G. Salaün and V. Vyatkin. 'Probabilistic Model Checking for IEC 61499: A Manufacturing Application'. In: ICIT 2024 - 25th IEEE International Conference on Industrial Technology. Bristol, United Kingdom: IEEE, 2024, pp. 1–6. URL: https://inria.hal.science/hal-04533520 (cit. on p. 18).

[16] I. Faqrizal, Q. Nivon and G. Salaün. 'Automated Repair of Violated Eventually Properties in Concurrent Programs'. In: FormaliSE 2024 - 12th IEEE/ACM International Conference on Formal Methods in Software Engineering. Lisbon, Portugal: IEEE, 2024, pp. 1–11. DOI: 10.1145/3644033.3644383. URL: https://inria.hal.science/hal-04566873 (cit. on p. 13).

[17] I. Faqrizal, G. Salaün and Y. Falcone. 'Guided Evolution of IEC 61499 Applications'. In: ETFA 2024 - 29th IEEE International Conference on Emerging Technologies and Factory Automation. Padova, Italy: IEEE, 2024, pp. 1–8. URL: https://inria.hal.science/hal-04680109 (cit. on p. 18).

[18] H. Garavel and B. Luttik. 'Four Formal Models of IEEE 1394 Link Layer'. In: *arXiv:2403.17862*. Proceedings of the 6th Workshop on Models for Formal Analysis of Real Systems (MARS 2O24). Luxembourg, Luxembourg: arXiv, 2024. DOI: 10.48550/arXiv.2403.18723. URL: https://inria.hal.science/hal-04776396 (cit. on p. 17).

[19] J.-B. Horel, A. Renzaglia, R. Mateescu and C. Laugier. 'Scenario-based Validation of Autonomous Vehicles using Augmented Reality'. In: ICRA 2024 - 40th Anniversary of the IEEE Conference on Robotics and Automation. Rotterdam, Netherlands, 2024, pp. 1–3. URL: https://hal.science/hal-04682329.

[20] P. Ledent, R. Mateescu and W. Serwe. 'Improving PSS Test Generation Using Model Checking and Conformance Testing'. In: FDL 2024 - 27th Forum on specification and Design Languages. Stockholm, Sweden: IEEE, 2024, pp. 1–9. DOI: 10.1109/FDL63219.2024.10673842. URL: https://inria.hal.science/hal-04719995 (cit. on p. 19).

[21] P. Ledent, R. Mateescu and W. Serwe. 'Testing Resource Isolation for System-on-Chip Architectures'. In: *Electronic Proceedings in Theoretical Computer Science*. MARS 2024 - 6th Workshop on Models for Formal Analysis of Real Systems. Vol. 399. Luxembourg, Luxembourg, 2024, pp. 1–40. DOI: 10.48550/arXiv.2403.18720. URL: https://inria.hal.science/hal-04573384 (cit. on p. 19).

[22] Q. Nivon and G. Salaün. 'Automated Generation of BPMN Processes from Textual Requirements'. In: ICSOC 2024 - 22nd International Conference on Service-Oriented Computing. Tunis, Tunisia, 2024, pp. 1–16. URL: https://inria.hal.science/hal-04734434 (cit. on p. 12).

[23] Q. Nivon and G. Salaün. 'Semi-Automated Refactoring of BPMN Processes'. In: QRS 2024 - IEEE 24th International Conference on Software Quality, Reliability, and Security. Cambridge, United Kingdom: IEEE, 2024, pp. 1–12. URL: https://inria.hal.science/hal-04644426 (cit. on p. 14).

Scientific book chapters

[24] H. Garavel, H. Hermanns and D. Parker. 'Revisiting a Pioneering Concurrent Stochastic Problem: The Erlangen Mainframe'. In: *Principles of Verification: Cycling the Probabilistic Landscape*. Vol. 15261. Lecture Notes in Computer Science. Springer Nature Switzerland, 13th Nov. 2024, pp. 46–74. DOI: 10.1007/978-3-031-75775-4_3. URL: https://inria.hal.science/hal-04909368 (cit. on p. 18).

Other scientific publications

[25] J.-B. Horel, A. Renzaglia, R. Mateescu and C. Laugier. *Scenario-based Validation of Autonomous Vehicles using Augmented Reality: Video of the Experiments*. 1st Mar. 2024. URL: https://hal.science/hal-04682356.

10.3 Cited publications

[26] D. Champelovier, X. Clerc, H. Garavel, Y. Guerte, C. McKinty, V. Powazny, F. Lang, W. Serwe and G. Smeding. 'Reference Manual of the LNT to LOTOS Translator (Version 6.8)'. INRIA, Grenoble, France. Jan. 2019 (cit. on p. 9).

[27] E. M. Clarke, E. A. Emerson and A. P. Sistla. 'Automatic Verification of Finite-State Concurrent Systems using Temporal Logic Specifications'. In: *ACM Transactions on Programming Languages and Systems* 8.2 (Apr. 1986), pp. 244–263 (cit. on p. 8).

[28] R. De Nicola and F. W. Vaandrager. 'Action versus State Based Logics for Transition Systems'. In: *Semantics of Concurrency*. Vol. 469. Lecture Notes in Computer Science. Springer Verlag, 1990, pp. 407–419 (cit. on p. 8).

[29] I. Faqrizal. 'Quantitative Verification and Runtime Techniques for Industrial Automation Systems'. Thèse de doctorat dirigée par Salaün, Gwen et Falcone, Yliès Carlo Informatique Université Grenoble Alpes 2024. PhD thesis. Université Grenoble Alpes, Dec. 2024. URL: http://www.theses.fr/s351716 (cit. on p. 18).

[30] H. Garavel. 'Compilation of LOTOS Abstract Data Types'. In: *Proceedings of the 2nd International Conference on Formal Description Techniques FORTE'89 (Vancouver B.C., Canada)*. Ed. by S. T. Vuong. North Holland, Dec. 1989, pp. 147–162 (cit. on p. 7).

[31] H. Garavel. 'OPEN/CÆSAR: An Open Software Architecture for Verification, Simulation, and Testing'. In: *Proceedings of the First International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS'98 (Lisbon, Portugal)*. Ed. by B. Steffen. Vol. 1384. Lecture Notes in Computer Science. Full version available as INRIA Research Report RR-3352. Berlin: Springer Verlag, Mar. 1998, pp. 68–84 (cit. on pp. 5, 7).

[32] H. Garavel and F. Lang. 'SVL: a Scripting Language for Compositional Verification'. In: *Proceedings of the 21st IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems FORTE'2001 (Cheju Island, Korea)*. Ed. by M. Kim, B. Chin, S. Kang and D. Lee. Full version available as INRIA Research Report RR-4223. IFIP. Kluwer Academic Publishers, Aug. 2001, pp. 377–392 (cit. on p. 9).

[33] H. Garavel, F. Lang and R. Mateescu. 'Compiler Construction using LOTOS NT'. In: *Proceedings of the 11th International Conference on Compiler Construction CC 2002 (Grenoble, France)*. Ed. by N. Horspool. Vol. 2304. Lecture Notes in Computer Science. Springer Verlag, Apr. 2002, pp. 9–13 (cit. on p. 9).

[34] H. Garavel, R. Mateescu and I. Smarandache-Sturm. 'Parallel State Space Construction for Model-Checking'. In: *Proceedings of the 8th International SPIN Workshop on Model Checking of Software SPIN'2001 (Toronto, Canada)*. Ed. by M. B. Dwyer. Vol. 2057. Lecture Notes in Computer Science. Revised version available as INRIA Research Report RR-4341 (December 2001). Berlin: Springer Verlag, May 2001, pp. 217–234 (cit. on p. 8).

[35] H. Garavel and W. Serwe. 'State Space Reduction for Process Algebra Specifications'. In: *Theoretical Computer Science* 351.2 (Feb. 2006), pp. 131–145 (cit. on p. 7).

[36] H. Garavel and J. Sifakis. 'Compilation and Verification of LOTOS Specifications'. In: *Proceedings of the 10th International Symposium on Protocol Specification, Testing and Verification (Ottawa, Canada)*. Ed. by L. Logrippo, R. L. Probert and H. Ural. IFIP. North Holland, June 1990, pp. 379–394 (cit. on p. 7).

[37] M. Hennessy and R. Milner. 'Algebraic Laws for Nondeterminism and Concurrency'. In: *Journal of the ACM* 32 (1985), pp. 137–161 (cit. on p. 8).

[38] H. Hermanns, U. Herzog and V. Merksiotakis. 'Stochastic Process Algebras as a Tool for Performance and Dependability Modelling.' In: *Proceedings of the International Computer Performance and Dependability Symposium (IPDS'95), Erlangen, Germany*. Apr. 1995, pp. 102–111 (cit. on p. 17).

[39] U. Herzog and V. Merksiotakis. 'Stochastic Process Algebras Applied to Failure Modelling'. In: *Proceedings of the 2nd Workshop on Process Algebras and Performance Modeling (PAPM'94), Regensberg/Erlangen, Germany*. July 1994, pp. 107–126 (cit. on p. 17).

[40] P. Ledent. 'Formal Modeling for Testing of System-on-Chip Resource Isolation'. Thèse de doctorat dirigée par Mateescu, Radu et Serwe, Wendelin Informatique Université Grenoble Alpes 2024. PhD thesis. Université Grenoble Alpes, Oct. 2024. URL: http://www.theses.fr/s274827 (cit. on p. 19).

[41] J. Magee and J. Kramer. *Concurrency: State Models and Java Programs*. 2006th ed. Wiley, Apr. 2006 (cit. on p. 9).

[42] L. Marsso, R. Mateescu and W. Serwe. 'Automated Transition Coverage in Behavioural Conformance Testing'. In: *ICTSS 2020 - 32nd IFIP International Conference on Testing Software and Systems*. Napoli, Italy, Dec. 2020, pp. 219–235. DOI: 10.1007/978-3-030-64881-7_14. URL: https://inria.hal.science/hal-03038050 (cit. on p. 19).

[43] L. Marsso, R. Mateescu and W. Serwe. 'TESTOR: A Modular Tool for On-the-Fly Conformance Test Case Generation'. In: *TACAS 2018 - 24th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Vol. 10806. Lecture Notes in Computer Science. Thessaloniki, Greece: Springer, Apr. 2018, pp. 211–228. DOI: 10.1007/978-3-319-89963-3_13. URL: https://hal.inria.fr/hal-01777861 (cit. on p. 16).

[44] R. Mateescu and D. Thivolle. 'A Model Checking Language for Concurrent Value-Passing Systems'. In: *Proceedings of the 15th International Symposium on Formal Methods FM'08 (Turku, Finland)*. Ed. by J. Cuellar, T. Maibaum and K. Sere. Vol. 5014. Lecture Notes in Computer Science. Springer Verlag, May 2008, pp. 148–164 (cit. on p. 4).

[45] M. Prédhumeau. 'Modélisation et simulation de comportements piétons réalistes en espace partagé avec un véhicule autonome'. Theses. Université Grenoble Alpes [2020-....], Dec. 2021. URL: https://hal.science/tel-03518751 (cit. on p. 16).

[46] M. Sighireanu and R. Mateescu. 'Verification of the Link Layer Protocol of the IEEE-1394 Serial Bus (FireWire): an Experiment with E-LOTOS'. In: *Springer International Journal on Software Tools for Technology Transfer (STTT)* 2.1 (Dec. 1998), pp. 68–88 (cit. on p. 17).

[47] A. Zuo. 'Modélisation, Analyse en Temps Réel et Vérification Quantitative des Processus Métier'. Thèse de doctorat dirigée par Salaün, Gwen et Falcone, Yliès Carlo Informatique Université Grenoble Alpes 2024. PhD thesis. Université Grenoble Alpes, Apr. 2024. URL: http://www.theses.fr/2024GRALM014 (cit. on p. 14).