IN PARTNERSHIP WITH:
Institut polytechnique de Grenoble

Université Joseph Fourier (Grenoble)

Activity Report 2017

Project-Team CONVECS

Construction of verified concurrent systems

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

RESEARCH CENTER
Grenoble - Rhône-Alpes

THEME
Proofs and Verification

Table of contents

Project-Team CONVECS

Creation of the Team: 2012 January 01, updated into Project-Team: 2014 January 01

Keywords:

Computer Science and Digital Science:

A1.1.6. - Cloud
A1.3. - Distributed Systems
A2.1.1. - Semantics of programming languages
A2.1.7. - Distributed programming
A2.4.1. - Analysis
A2.4.2. - Model-checking
A2.5. - Software engineering
A5.11.1. - Human activity analysis and recognition
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:

B6.1.1. - Software engineering
B6.3.2. - Network protocols
B6.4. - Internet of things
B6.6. - Embedded systems
B8.1. - Smart building/home

1. Personnel

Research Scientists

Radu Mateescu [Team leader, Inria, Senior Researcher, HDR]
Hubert Garavel [Inria, Senior Researcher]
Frédéric Lang [Inria, Researcher]
Wendelin Serwe [Inria, Researcher]

Faculty Member

Gwen Salaün [UGA, Professor, HDR]

PhD Students

Gianluca Barbon [UGA]
Lina Marsso [Inria]
Ajay Muroor Nadumane [Inria, from Oct 2017]
Umar Ozeer [Orange Labs]

Technical staff

Lian Apostol [Inria, from Dec 2017]

Interns

Jean-Philippe Gros [Inria, from Feb 2017 until Jul 2017]
Waqas Imtiaz [Inria, from Feb 2017 until Jul 2017]
Ajay Muroor Nadumane [Inria, from Feb 2017 until Jul 2017]

Julie Parreaux [Inria, from May 2017 until Jul 2017]

Administrative Assistant

Myriam Etienne [Inria]

Visiting Scientist

Soren Enevoldsen [Aalborg University, from Sep 2017 until Dec 2017]

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 [24] 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*) [47], 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

4.1. 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,
- *Databases:* transaction protocols, distributed knowledge bases, stock management,
- *Distributed systems:* virtual shared memory, dynamic reconfiguration algorithms, fault tolerance algorithms, cloud computing,
- *Embedded systems:* air traffic control, avionic systems, medical devices,
- *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 and Platforms

5.1. CADP Pro

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*) [4] 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 [40], CTL [26], 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 [46] 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 the VERTECS project-team at 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.

- Participants: Frédéric Lang, Hubert Garavel, Radu Mateescu and Wendelin Serwe
- Contact: Hubert Garavel
- URL: http://cadp.inria.fr/

5.2. 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.

- Participants: Frédéric Lang, Hubert Garavel and Wendelin Serwe
- Contact: Hubert Garavel
- URL: http://convecs.inria.fr/software/traian/

6. New Results

6.1. New Formal Languages and their Implementations

LNT is a next generation formal description language for asynchronous concurrent systems, which attempts to combine the best features of imperative programming languages and value-passing process algebras. LNT is increasingly used by CONVECS for industrial case studies and applications (see § 6.5) and serves also in university courses on concurrency, in particular at ENSIMAG (Grenoble) and at Saarland University.

6.1.1. Translation from LNT to LOTOS

Participants: Hubert Garavel, Frédéric Lang, Wendelin Serwe.

The move towards "safer" LNT exceptions initiated in 2016 has been completed in 2017: the two concepts of gates and exceptions have been unified in both LNT processes and LNT functions. The static semantics of LNT no longer requires that variables and exceptions share the same name space.

LNT now permits simple loops (of the form "loop ... end loop", without loop label nor "while" condition) in LNT functions, as well as in LNT processes.

The pragma names "comparedby", "external", "implementedby", "iteratedby", "printedby", and "representedby" are no longer reserved LNT keywords, meaning that it is now permitted to declare LNT identifiers having these names. Two new type pragmas "!card" and "!bits" have been added to specify the maximum number of values and the number of bits to be used when storing the values of a given type in "hash-consing" tables.

The LPP preprocessor and the LNT2LOTOS translator, which implement the LNT language, have been enhanced in many ways. In addition to 9 bug fixes, the following enhancements have been made:

- LPP now implements LNT character strings more concisely.
- LPP automatically adds the ".lnt" extension to input and output files if this extension is missing.
- The algorithm that computes which LNT gates are used in each function or process has been made more precise, and LNT2LOTOS now warns about gates that are declared but never used.
- LNT2LOTOS performs stricter compile-time checks that produce dedicated error messages, rather than generating invalid LOTOS code that was subsequently rejected by CAESAR and/or CAESAR.ADT. Also, several error messages displayed by LNT2LOTOS during its static-analysis phases have been enhanced.
- The translation from LNT functions to LOTOS operations has been significantly improved by eliminating unreachable or redundant LOTOS equations, removing unused auxiliary LOTOS operations, simplifying the premises of certain LOTOS equations, factorizing identical assignments in "if-then-else" instructions, and optimizing long sequences of assignments intertwined with assertions. Thus, LNT2LOTOS is now faster, uses less memory, generates more compact LOTOS code, and can compile larger LNT specifications that could not be handled before.

The LNT2LOTOS Reference Manual, which contains the definition of the LNT language, has been revised, enriched, and simplified in many ways. A paper presenting the historical background and motivation behind the definition of LNT was published in an international conference [18].

6.1.2. NUPN

Participant: Hubert Garavel.

Nested-Unit Petri Nets (NUPNs) is an upward-compatible extension of P/T nets, which are enriched with structural information on their concurrent structure. Such additional information can easily be produced when NUPNs are generated from higher-level specifications (e.g., process calculi); quite often, such information allows logarithmic reductions in the number of bits required to represent states, thus enabling verification tools to perform better. The principles of NUPNs are exposed in [29] and its PNML representation is described here [1].

In 2017, we studied an abstraction called *place fusion*, which takes advantage of the compositional, hierarchical structure of NUPNs. We formulated key theorems stating which properties are preserved or not under this abstraction. On the practical side, our collection of NUPN models grew to more than 8 000 benchmarks. Statistical studies were done on this collection to estimate the compression factor permitted by the NUPN model. A journal article providing an overview of NUPNs was written.

The NUPN model has been adopted by the Model Checking Contest and implemented in ten different tools developed in four countries. In 2017, the NUPN model was also adopted for the parallel problems of the RERS'2017 (*Rigorous Examination of Reactive Systems*) challenge [2].

[1] http://mcc.lip6.fr/nupn.php

6.1.3. Analysis of BPMN via Translation to LNT

Participants: Ajay Muroor Nadumane, Gwen Salaün.

Business process modeling is an important concern in companies and organizations. Formal analysis techniques are crucial to detect semantic issues in the corresponding models, or to help with their refactoring and evolution. However, business process development frameworks often fall short when it comes to go beyond simulation or syntactic checking of the models. To ensure a more robust development of business processes, we developed the VBPMN verification framework. It features several techniques for the automatic analysis of business processes modeled using BPMN, the de facto standard for business process modeling.

The business processes, described using a Web application compliant with BPMN 2.0, are transformed into an intermediate format called PIF (*Process Intermediate Format*). Then, from the PIF descriptions, models in LNT and model-specific verification scripts in SVL are generated. In the end, CADP is used to check either for functional properties of a given business process, or for the correctness of the evolution of a business process into another one. This latter kind of verification supported by VBPMN is particularly helpful in order to improve a process w.r.t. certain optimization criteria. A paper presenting these results was published in an international conference [16].

6.1.4. Translation of Term Rewrite Systems

Participants: Hubert Garavel, Lina Marsso.

We pursued the development undertaken in 2015 of a software platform for systematically comparing the performance of rewrite engines and pattern-matching implementations in algebraic specification and functional programming languages. Our platform reuses the benchmarks of the three Rewrite Engine Competitions (2006, 2009, and 2010). Such benchmarks are term-rewrite systems expressed in a simple formalism named REC, for which we developed automated translators that convert REC benchmarks into many languages, among which AProVE, Clean, Haskell, LNT, LOTOS, Maude, mCRL, MLTON, OCAML, Opal, Rascal, Scala, SML-NJ, Stratego/XT, and Tom.

In 2017, we revised and enhanced the largest REC benchmark, the MAA (*Message Authenticator Algorithm*), a Message Authentication Code used for financial transactions (ISO 8731-2) between 1987 and 2002. This model (13 sorts, 18 constructors, 644 non-constructors, and 684 rewrite rules) was proven to be confluent, and terminating. Implementations in thirteen different languages have been automatically derived from this model and used to validate 200 official test vectors for the MAA. These results led to a publication in an international conference [14].

We also corrected and/or enhanced several of the existing REC translators (e.g., Clean) and added support of CafeOBJ and compiled OCAML. A scientific paper on this study has been prepared.

6.1.5. Other Language Developments

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

The ability to compile and verify formal specifications with complex, user-defined operations and data structures is a key feature of the CADP toolbox since its very origins.

In 2017, we brought various enhancements to several compilers handling formal specification languages (LOTOS, MCL, XTL, and GRL):

- A buffer overflow and two out-of-bound array accesses have been corrected in both CAESAR and CAESAR.ADT. Two memory allocation bugs have also been corrected in CAESAR.ADT. The latter tool now generates C code that gives better diagnostic when the evaluation of a constant fails at run time (e.g., when it triggers an exception signal, or exhausts the stack or heap memory).
- In addition to two bug fixes, the warning and error messages displayed by MCL_EXPAND and XTL_EXPAND have been made more precise and stringent. The XTL libraries “walk” and “walk_nice” have been modified not to trigger the extra warnings recently introduced.

[2] http://www.rers-challenge.org

- The GRL2LNT translator takes as input a formal description in GRL of a GALS system and generates an equivalent LNT specification. A new version 1.1 of GRL2LNT has been released, which corrects a bug concerning the LNT code generated by the "`-merge`" option.

H. Garavel pursued the study of the most suitable axiomatization of signed integers undertaken in 2016. He reviewed a tenth of such Peano-like axiom systems, which he classified and evaluated according to complexity and efficiency criteria. These results have been published in an international conference [13].

6.2. Parallel and Distributed Verification

6.2.1. Distributed State Space Manipulation

Participants: Hubert Garavel, Wendelin Serwe.

For distributed verification, CADP provides the PBG format, which implements the theoretical concept of *Partitioned LTS* [34] and provides a unified access to an LTS distributed over a set of remote machines.

In 2017, many changes have been done to simplify the code of the CAESAR_NETWORK_1 communication library, which is the backbone of the distributed verification tools of CADP, as well as the code of other tools such as BCG_MIN, but most of these changes are not directly observable by end users. In addition to two bug fixes in CAESAR_NETWORK_1 and two other bug fixes in the BES_SOLVE tool, the error messages displayed by the various tools and the statistical information produced by the "`-stat`" option of BES_SOLVE have been made more concise and more informative.

6.2.2. Debugging of Concurrent Systems

Participants: Gianluca Barbon, Gwen Salaün.

Model checking is an established technique for automatically verifying that a model satisfies a given temporal property. When the model violates the property, the model checker returns a counterexample, which is a sequence of actions leading to a state where the property is not satisfied. Understanding this counterexample for debugging the specification is a complicated task for several reasons: (i) the counterexample can contain hundreds of actions, (ii) the debugging task is mostly achieved manually, and (iii) the counterexample does not explicitly highlight the source of the bug that is hidden in the model.

We proposed an approach that improves the usability of model checking by simplifying the comprehension of counterexamples. Our solution aims at keeping only actions in counterexamples that are relevant for debugging purposes. To do so, we first extract in the model all the counterexamples. Second, we define an analysis algorithm that identifies actions that make the model skip from incorrect to correct behaviours, making these actions relevant from a debugging perspective. Our approach is fully automated by a tool we implemented and applied on real-world case studies from various application areas for evaluation purposes. This work led to a publication in an international conference [11].

In 2017, we focused on extending our approach following three directions: (a) we introduced new notions to identify new types of relevant actions; (b) we developed a set of heuristics to extract these actions from counterexamples; (c) we proposed an alternative approach to focus on a broader range of properties (i.e., liveness properties). These new extensions have been integrated into our tool. A paper was submitted to an international journal.

6.3. Timed, Probabilistic, and Stochastic Extensions

6.3.1. Tools for Probabilistic and Stochastic Systems

Participants: Hubert Garavel, Jean-Philippe Gros, Frédéric Lang, Julie Parreaux, Wendelin Serwe.

Formal models and tools dealing with quantititative aspects (such as time, probabilities, and other continuous physical quantities) have become unavoidable for a proper study and computer-aided verification of functional and non-functional properties of cyberphysical systems. The wealth of such formal models is sometimes referred to as a quantitative "zoo" [39].

The CADP toolbox already implements some of these probabilistic/stochastic models, namely DTMCs and CTMCs (*Discrete-Time* and *Continuous-Time Markov Chains*), and IMCs (*Interactive Markov Chains*) [41]. Our long-term goal is to increase the capability and flexibility of the CADP tools, so as to support other quantitative models more easily.

In 2017, we undertook a systematic review of the existing theoretical models and built a comprehensive list of more than 70 software tools implementing these models [37]. The results of this study have been made widely available as a Web catalog [3].

In parallel, we also undertook a systematic review [50] of the benchmarks made available for these tools. We downloaded more than 21 000 files from the web and developed triage scripts to analyze these files and classify them automatically, separating various kinds of automata-based models (e.g., Markov chains, Markov automata, hybrid automata, etc.) from temporal-logic formulas. One finding of this "big data" study is the present lack of diversity, as four tools (PRISM, MRMC, STORM, and SiSAT) provide nearly 60% of the models.

To address this issue, we started investigating the probabilistic and stochastic models of complex industrial systems produced by former PhD students of the VASY and CONVECS teams. We analyzed these models (written in BCG, EXP, LOTOS, LNT, SVL, and/or Makefiles) to separate functional aspects from performance ones, leading to a collection of DTMCs, CTMCs, IMCs, and IPCs (*Interactive Probabilistic Chains*). We updated these models to ensure compatibility with the latest versions of CADP and C compilers, and we started enhancing EXP.OPEN with new features that simplify the parallel composition of IPCs (see § 6.4.1).

6.3.2. On-the-fly Model Checking for Extended Regular Probabilistic Operators

Participant: Radu Mateescu.

Specifying and verifying quantitative properties of concurrent systems requires expressive and user-friendly property languages combining temporal, data-handling, and quantitative aspects. In collaboration with José Ignacio Requeno (Univ. Zaragoza, Spain), we undertook the quantitative analysis of concurrent systems modeled as PTSs (*Probabilistic Transition Systems*), whose actions contain data values and probabilities. We proposed a new regular probabilistic operator that extends naturally the Until operators of PCTL (*Probabilistic Computation Tree Logic*) [38], by specifying the probability measure of a path characterized by a generalized regular formula involving arbitrary computations on data values. We integrated the regular probabilistic operator into MCL, we devised an associated on-the-fly model checking method based on a combined local resolution of linear and Boolean equation systems, and we implemented the method in a prototype extension of the EVALUATOR model checker.

In 2017, we continued experimenting the extended model checker on further examples of protocols (Bounded Retransmission Protocol, randomized philosophers, self-stabilization) and observed that it exhibits a performance comparable with the explicit-state algorithms of the PRISM model checker [4]. A paper was submitted to an international journal.

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

6.4.1. Compositional Verification

Participants: Hubert Garavel, Frédéric Lang.

The CADP toolbox contains various tools dedicated to compositional verification, among which EXP.OPEN, BCG_MIN, BCG_CMP, and SVL play a central role. EXP.OPEN explores on the fly the graph corresponding to a network of communicating automata (represented as a set of BCG files). BCG_MIN and BCG_CMP respectively minimize and compare behavior graphs modulo strong or branching bisimulation and their stochastic extensions. SVL (*Script Verification Language*) is both a high-level language for expressing complex verification scenarios and a compiler dedicated to this language.

[3] http://cadp.inria.fr/resources/zoo
[4] http://www.prismmodelchecker.org/

In 2017, two bugs have been solved in SVL and one bug has been solved in EXP.OPEN. Several improvements have been brought to both tools. In particular:

- EXP.OPEN now has two new options "`-prob`" and "`-rate`" for handling probabilistic and stochastic transitions, respectively; without these options, probabilistic and stochastic transitions are considered as ordinary transitions (this enables EXP.OPEN to be used for implementing alternative semantics, such as *Interactive Probabilistic Chains* [27] where probabilistic transitions are synchronized using a global clock). Consequently, the former "`-ratebranching`" option has been replaced by "`-rate -branching`".

  Also, error messages about synchronization vectors have been made more precise and EXP.OPEN performs tighter checks about labels containing only blanks and unexpected synchronization of probabilistic or stochastic transitions. Two bugs have been fixed in EXP.OPEN and style files have been added to bring support for the EXP format by mainstream text editors.
- A new option "`-v`" has been added to set SVL variables from the command line (similar to "awk" or "make"). Debugging SVL scripts has been made easier: the "`-debug`" option of SVL now stops the execution as soon as a shell command (e.g., a CADP tool or a Unix command) terminates with a non-zero exit status, so that problems are detected as soon as they occur.

  Also, SVL now performs tighter semantic checks, making sure that all partial-order reduction options passed to EXP.OPEN (namely, options explicitly set by the user and options automatically computed by SVL from the context of the EXP composition expression) are not contradictory.

6.4.2. On-the-Fly Test Generation

Participants: Hubert Garavel, Lina Marsso, Radu Mateescu, Wendelin Serwe.

The CADP toolbox provides support for conformance test case generation by means of the TGV tool. Given a formal specification of a system and a test purpose described as an input-output LTS (IOLTS), TGV automatically generates test cases, which assess using black box testing techniques the conformance of a system under test w.r.t. the formal specification. A test purpose describes the goal states to be reached by the test and enables one to indicate parts of the specification that should be ignored during the testing process. TGV does not generate test cases completely on the fly (i.e., *online*), because it first generates the complete test graph (CTG) and then traverses it backwards to produce controllable test cases.

In 2017, we carried out the following activities:

- We developed the prototype tool TESTOR to extract test cases completely on the fly. Compared to TGV, the new tool TESTOR presents several advantages: (i) it has a more modular architecture, based on generic graph transformation components taken from the OPEN/CAESAR libraries (τ-compression, τ-confluence, τ-closure, determinization, resolution of Boolean equation systems); (ii) it is capable of extracting a test case completely on the fly, by exploiting the diagnostic generation features of the Boolean equation system resolution algorithms; (iii) it enables a more flexible expression of test purposes, taking advantage of the multiway rendezvous, a primitive to express communication and synchronization among a set of distributed processes [15]. We evaluated TESTOR on three published case studies and more than 10 000 examples taken from the non-regression test suites of CADP. A paper describing this work was accepted for publication in an international conference.
- We also revised TGV, which is now by default much less verbose and only displays the most important information, but the former behaviour can still be retained using option "`-verbose`". A new option "`-monitor`" allows to follow in real time how the test case generation progresses. Many warning and error messages have been enhanced, various bugs (especially buffer overflows) have been fixed, and memory allocation results are now strictly controlled.

6.4.3. Other Component Developments

Participants: Lian Apostol, Soren Enevoldsen, Hubert Garavel, Frédéric Lang, Radu Mateescu, Wendelin Serwe.

The CAESAR_STANDARD library was enriched with the new CAESAR_TYPE_FORMAT type and its associated primitives, and with two new functions CAESAR_SET_SIGNALS() and CAESAR_RESET_SIGNALS() for handling POSIX signals (including SIGSEGV, i.e., segmentation violation). The CAESAR_GRAPH interface, which remained stable for two decades, has been modified: its two functions CAESAR_FORMAT_STATE() and CAESAR_FORMAT_LABEL() became more powerful, while its two functions CAESAR_MAX_FORMAT_STATE() and CAESAR_MAX_FORMAT_LABEL() have been removed from the interface. The same changes apply as well to all the other similar functions of the OPEN/CAESAR libraries. All the OPEN/CAESAR compilers, application tools, and demo examples have been modified to reflect these changes.

Sustained effort has been made to ensure that CADP works properly on mainstream computing platforms. In particular, the RFL and TST scripts and the documentation have been continuously updated. Changes were brought to CADP to cope with recent C compilers (such as GCC 6 and Clang) and to work around problems with the "indent" command available on Solaris and macOS/Xcode. On Linux, CADP was ported to the latest versions of Centos, Debian 9, and Ubuntu 17.04. The support for the various desktop environments (Gnome, KDE, Mate, etc.) available in Linux distributions has improved. On macOS, support of obsolete versions (from Mac OS X 10.6 "Snow Leopard" to OS X 10.9 "Mavericks" included) was withdrawn and support of macOS 10.13 "High Sierra" was added. Preliminary steps have been made to prepare a 64-bit version of CADP on macOS. On Windows, support of obsolete versions (Windows XP and Vista) was dropped. CADP was also adapted to follow the changes in the Cygwin software regarding pipe management. Many changes were made to CADP so as to support the case where Cygwin is not installed in "`C:/`" but in a different folder. Finally, preliminary steps have been made towards a 64-bit version of CADP for Windows.

In collaboration with Soren Enevoldsen (Aalborg University, Denmark), we studied the applicability of CADP tools for analyzing concurrent systems described using weighted CCS (WCCS) [43], an extension of CCS with an action prefix operator carrying a weight represented as a natural number. We developed a prototype OPEN/CAESAR-compliant compiler for WCCS, which enables to produce, in conjunction with the GENERATOR tool of CADP, the corresponding LTS model in which transitions are labeled with actions and weights. For specifying temporal properties of WCCS systems, we developed a prototype MCL library defining the operators of weighted CTL (WCTL) [43] using MCL fixed point operators parameterized by natural numbers. This library, used in conjunction with the EVALUATOR tool, provides an on-the-fly model checker for WCTL equipped with diagnostic capabilities (counterexamples and witnesses).

6.5. Real-Life Applications and Case Studies

6.5.1. Autonomous Resilience of Distributed IoT Applications in a Fog Environment

Participants: Umar Ozeer, Gwen Salaün.

The first year of the PhD thesis started with a state of the art on the resilience mechanisms, broadly in distributed systems and then more specifically in distributed IoT (*Internet of Things*) applications. This resulted, firstly in defining the scope of the thesis and, secondly, in identifying the steps to manage failures, namely state saving, failure detection, fault isolation, and repairing in a consistent state.

A study of the mechanisms for saving the execution state of processes in distributed systems was done. This enabled us to identify the specificities of our environment and to adapt existing snapshot and message logging mechanisms to fit the context of state saving and manipulation in distributed IoT applications in view of repairing failures and re-establishing consistency. We devised a first failure management protocol, which is being tested on an instance of an IoT application test bed at Orange Labs. Next steps include formally verifying the failure management protocol, as well as carrying out further tests on larger scaled applications for the purpose of performance evaluation.

6.5.2. Activity Detection in a Smart Home

Participants: Waqas Imtiaz, Frédéric Lang, Radu Mateescu, Wendelin Serwe.

Ambient intelligence is an active research field, whose aim is to design and analyze smart environments that are capable of automated interaction with users and the physical world, through sensors, actuators, displays, and computational elements, embedded in everyday objects, and connected through a network. In the Grenoble area, the Equipex Amiqual4Home [5] provides among others access to a Smart Home, which is a fully functional two-stage 90 meters square flat with 4 rooms including an open to kitchen living room, a bedroom, a bathroom and a small office. All the rooms are equipped with cameras, microphones, sensors and actuators to remote control various appliances like rollershutter, lights or multimedia devices. The software architecture of the Smart Home is based on the open source home automation software OpenHAB [6]. It allows a complete control of the flat devices with a single system, despite the various protocols used. Using the rule engine, it also enables the definition of rules expressing how the Smart Home should react to physical (human action, sensors, etc.) or external (weather prediction service, calendar, etc.) events. A difficult question is how to make sure that smart environments are programmed correctly, and will not lead to unexpected or even harmful behaviour.

Smart environments are concurrent and asynchronous by nature. To address the question above, we started, in collaboration with Nicolas Bonnefond (PERVASIVE INTERACTION team and Amiqual4Home), to study how existing tools for the formal design and verification of concurrent asynchronous systems present in the CADP toolbox can be used to verify a smart environment. Firstly, we proposed a translation from OpenHAB rules into a formal LNT model on which properties can be verified [42]. Secondly, in collaboration with Paula Lago and Claudia Roncancio (SIGMA team of LIG), we exploited the dataset ContextAct@A4H of daily living activities collected and annotated within Amiqual4Home for the purpose of activity recognition. Each activity was described as an MCL temporal logic formula that is checked repeatedly on the log of sensor measurements until all occurrences of the activity have been found. This approach has the ability to recognize the start and end points of activities (thus not requiring to segment sensor data) and also expresses the temporal order of events, thus palliating a limitation of existing ontology based activity recognition techniques. This led to a publication in an international conference [17].

6.5.3. Other Case Studies

Participants: Hubert Garavel, Frédéric Lang, Lina Marsso, Wendelin Serwe.

The demo examples of CADP, which have been progressively accumulated since the origins of the toolbox, are a showcase for the multiple capabilities of CADP, as well as a test bed to assess the new features of the toolbox. In 2017, the effort to maintain and enhance these demos has been pursued. The demo 05 (Airplane-ground communication protocol) has been modified to use the new syntax of exceptions in the LNT language. The LOTOS and LNT specifications of demo 12 (Message Authenticator Algorithm) have been entirely revised, based upon the fine knowledge acquired by modelling this cryptographic function as a term rewrite system [14]. The LNT specification has also been extended to incorporate the test vectors given in the International Standards ISO 8730 and 8731-2. The resulting specification, which was initially too large to be compiled, is now successfully handled after the enhancements brought to the LNT2LOTOS translator. Demo 19 (Production Cell) has been simplified and is now fully documented in a publication [15].

In the framework of the SECURIOT-2 project (see § 8.2.2.1), a Memory Protection Unit has been formally specified in LNT and verified at Tiempo using CADP. A paper has been submitted to an international conference.

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Grants with Industry

7.1.1. Orange Labs

Participants: Umar Ozeer, Gwen Salaün.

[5] http://amiqual4home.inria.fr
[6] http://www.openhab.org

Umar Ozeer is supported by a PhD grant (from November 2016 to November 2019) from Orange Labs (Grenoble) on detecting and repairing failures of data-centric applications distributed in the cloud and the IoT (see § 6.5.1), under the supervision of Xavier Etchevers (Orange Labs), Gwen Salaün (CONVECS), François Gaël Ottogalli (Orange Labs), and Jean-Marc Vincent (POLARIS project-team).

7.1.2. Nokia Bell Labs

Participants: Radu Mateescu, Ajay Muroor Nadumane, Gwen Salaün.

Ajay Muroor Nadumane is supported by a PhD grant (from October 2017 to October 2020) from Nokia Bell Labs (Nozay) on IoT service composition supported by formal methods, under the supervision of Gwen Salaün (CONVECS), Radu Mateescu (CONVECS), Ludovic Noirie, and Michel Le Pallec (Nokia Bell Labs).

8. Partnerships and Cooperations

8.1. Regional Initiatives

8.1.1. ARC6 Programme

Participants: Lina Marsso, Radu Mateescu [correspondent], Wendelin Serwe.

ARC6 is an academic research community funded by the Auvergne Rhône-Alpes region, whose objective is to foster the scientific collaborations between different academic institutions of the region working in the domain of information and communication technologies. ARC6 organizes various scientific animations (conferences, working groups, summer schools, etc.) and issues a yearly call for PhD and post-doctorate research project proposals.

Lina Marsso is supported by an ARC6 grant (from October 2016 to October 2019) on formal methods for testing networks of programmable logic controllers, under the supervision of Radu Mateescu and Wendelin Serwe (CONVECS), Ioannis Parissis and Christophe Deleuze (LCIS, Valence).

8.2. National Initiatives

8.2.1. PIA (Programme d'Investissements d'Avenir)

8.2.1.1. CAPHCA

Participants: Frédéric Lang, Radu Mateescu [correspondent], Wendelin Serwe.

CAPHCA (*Critical Applications on Predictable High-Performance Computing Architectures*) is a project funded by the PIA. The project, led by IRT Saint-Exupéry (Toulouse), involves a dozen of industrial partners (among which Airbus, CS Systèmes d'Information, Synopsis, and Thalès Avionics), the University Paul Sabatier (Toulouse), and Inria Grenoble – Rhône-Alpes (CONVECS and SPADES project-teams). CAPHCA addresses the dual problem of achieving performance and determinism when using new, high performance, multicore System-on-Chip (SoC) platforms for the deployment of real-time, safety-critical applications. The methodology adopted by CAPHCA consists in building a pragmatic combination of methods, tools, design constraints and patterns deployable at a short-term horizon in the industrial domains targeted in the project.

CAPHCA started in December 2017 for four years. The main contributions of CONVECS to CAPHCA are the detection of concurrency errors in parallel applications by means of formal methods and verification techniques.

8.2.2. Competitivity Clusters

8.2.2.1. SECURIOT-2

Participants: Lian Apostol, Hubert Garavel [correspondent], Radu Mateescu, Wendelin Serwe.

SECURIOT-2 is a project funded by the FUI (*Fonds Unique Interministériel*) within the *Pôle de Compétitivité* Minalogic. The project, led by Tiempo Secure (Grenoble), involves the SMEs (*Small and Medium Enterprises*) Alpwise, Archos, Sensing Labs, and Trusted Objects, the Institut Fourier and the VERIMAG laboratories of Université Grenoble Alpes, and CONVECS. SECURIOT-2 aims at developing a secure micro-controller unit (SMCU) that will bring to the IoT a high level of security, based on the techniques used for smart cards or electronic passports. The SMCU will also include an original power management scheme adequate with the low power consumption constraints of the IoT.

SECURIOT-2 started in September 2017 for three years. The main contributions of CONVECS to SECURIOT-2 are the formal modeling and verification of the asynchronous hardware implementing the secure elements developed by the project partners.

8.2.3. Other National Collaborations

We had sustained scientific relations with the following researchers:

- Pierre Boullier (Inria, team ALPAGE),
- Anne-Lise Courbis (Ecole des Mines, Alès, France),
- Christophe Deleuze and Ioannis Parissis (LCIS, Valence),
- Xavier Etchevers (Orange Labs, Meylan),
- Laurent Georget (Centrale/Supelec, Rennes, France),
- Claude Girault (LIP6, Paris),
- Fabrice Kordon and Lom Messan Hillah (LIP6, Paris),
- Xavier Leroy (Inria, team GALLIUM),
- Pascal Poizat (LIP6, Paris).

8.3. European Initiatives

8.3.1. Collaborations in European Programs, Except FP7 & H2020

Program: PHC Amadeus

Project acronym: RIDINGS

Project title: Rigourous Development of GALS Systems

Duration: January 2017 – December 2018

Coordinator: Inria Grenoble – Rhône-Alpes / CONVECS

Other partners: TU Graz, Institute of Software Technology (Austria)

Abstract: GALS systems, composed of synchronous components (driven by local clocks) that communicate through a network, are increasingly spreading with the development of the IoT. GALS systems are intrinsically complex due to the interplay of synchronous and asynchronous aspects, which make their development and debugging difficult. Therefore, it is necessary to adopt rigorous design methodologies, based on formal methods assisted by efficient validation tools. The RIDINGS project aims at enhancing the design flow of a GALS system by integrating the automatic generation of conformance tests from the formal model and the temporal properties used for verifying the system. This yields a double benefit for the designer: (i) it makes possible to check that a physical implementation conforms to the verified model; (ii) the development cost of the model and properties is distributed on the verification and testing phases of the design process, therefore increasing the return on investment.

8.3.2. Collaborations with Major European Organizations

The CONVECS project-team is member of the FMICS (*Formal Methods for Industrial Critical Systems*) working group of ERCIM [7]. H. Garavel and R. Mateescu are members of the FMICS board, H. Garavel being in charge of dissemination actions.

[7] http://fmics.inria.fr

8.4. 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.

8.4.1. Inria International Partners

8.4.1.1. Informal International Partners

Saarland University (Germany): we collaborate on a regular basis with the DEPEND (*Dependable Systems and Software*) research group headed by Holger Hermanns, who received an ERC Advanced Grant ("POWVER") in 2016.

8.4.2. Other International Collaborations

In 2017, we had scientific relations with several universities and institutes abroad, including:

- University of Málaga, Spain (Francisco Duran),
- University of Boumerdes, Algeria (Sarah Chabane),
- Saarland University, Germany (Alexander Graf-Brill),
- ISTI/CNR, Pisa, Italy (Franco Mazzanti),
- FBK, Torino, Italy (Gianni Zampedri),
- RWTH Aachen, Germany (Christian Dehnert),
- University of Twente, The Netherlands (Enno Ruijters),
- University of York, UK (Jan Staunton),
- University Rio Grande do Norte, Brazil (Wellison Raul Mariz Santos),
- University of Cali, Colombia (Camilo Rocha),
- Utah State University, USA (Nazmus Sakib and Zhen Zhang).

8.5. International Research Visitors

8.5.1. Visits of International Scientists

- Mahsa Shirmohammadi (University of Oxford, UK) visited us on February 23–24, 2017. She gave a talk on February 24, entitled "*Minimal Probabilistic Automata have to make Irrational Choices*".
- Josip Bozic, Birgit Hofer, Hermann Felbinger, and Franz Wotawa (TU Graz, Austria) visited us from May 15 to May 19, 2017, and attended the 1st RIDINGS Workshop held on May 17, 2017 at Inria Grenoble – Rhône-Alpes. J. Bozic gave a talk entitled "*Security Testing Based on Attack Patterns and Planning*". B. Hofer gave a talk entitled "*Fault Localization in Software and Spreadsheets*". H. Felbinger gave a talk entitled "*Test-Suite Reduction Does Not Necessarily Require Executing The Program Under Test*". F. Wotawa gave a talk entitled "*Research Activities at the Institute for Software Technology / TU Graz*".
- Soren Enevoldsen (Aalborg University, Denmark) visited us from September 27 to December 27, 2017. He gave a talk entitled "*Parallel Model Checking and Quantitative Models*" on October 24, 2017.

8.5.2. Visits to International Teams

- H. Garavel is an invited professor at Saarland University (Germany) as a holder of the Gay-Lussac Humboldt Prize.
- G. Salaün visited the University of Málaga (Spain) from May 31 to June 14, 2017.
- L. Marsso and W. Serwe visited TU Graz (Austria) from November 13 to November 17, 2017 in the framework of the PHC RIDINGS project.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific Events Organisation

9.1.1.1. Member of the Organizing Committees

- H. Garavel is a member of the model board [8] of MCC (*Model Checking Contest*). In 2017, he helped preparing new models (especially those in the NUPN format) and verified, using the CÆSAR.BDD tool of CADP, the forms describing all benchmark models submitted by the contest participants; this revealed a number of inconsistencies. The results of MCC'2017 have been published online [45] and a journal paper is in preparation.
- Together with Peter Höfner (Data61, CSIRO, Sydney, Australia), H. Garavel set up a model repository (hosted on the Gforge of Inria) to collect and archive formal models of real systems; this infrastructure is used by the series of MARS workshops [9]. This repository currently contains 17 models, two of which (a Message Authenticator Algorithm and a Production Cell) were deposited in 2017 by CONVECS.
- G. Salaün is member of the steering committee of the SEFM (*International Conference on Software Engineering and Formal Methods*) conference series since 2014.
- G. Salaün is member of the steering committee of the FOCLASA (*International Workshop on Foundations of Coordination Languages and Self-Adaptative Systems*) workshop series since 2011.

9.1.2. Scientific Events Selection

9.1.2.1. Chair of Conference Program Committees

- G. Barbon was publicity chair of FOCLASA'2017 (*15th International Workshop on Foundations of Coordination Languages and Self-Adaptative Systems*) co-located with SEFM'2017 (*15th International Conference on Software Engineering and Formal Methods*), Trento, Italy, September 5, 2017.
- F. Lang was co-chair of the Formal Methods track of ETR'2017 (*9ème Ecole d'été Temps-Réel*), Paris, France, August 28 - September 1, 2017.
- R. Mateescu was tutorial chair of QRS'2017 (*IEEE International Conference on Software Quality, Reliability, and Security*), Prague, Czech Republic, July 25–29, 2017.
- G. Salaün was co-chair of FOCLASA'2017.

9.1.2.2. Member of the Conference Program Committees

- H. Garavel was program committee member of the 7th FMF (*Forum Methodes Formelles*), Toulouse-Grenoble-Saclay-Rennes, France, January 31, 2017.
- H. Garavel was program committee member of MARS'2017 (*2nd Workshop on Models for Formal Analysis of Real Systems*), Uppsala, Sweden, April 29, 2017.
- H. Garavel and G. Salaün were program committee members of SEFM'2017 (*15th International Conference on Software Engineering and Formal Methods*), Trento, Italy, September 6–10, 2017.
- H. Garavel was program committee member of the 8th FMF (*Forum Methodes Formelles*), Toulouse-Grenoble-Saclay-Rennes, France, October 10, 2017.
- F. Lang was program committee member of GaM'2017 (*3rd Workshop on Graphs as Models*), Uppsala, Sweden, April 22–23, 2017.
- R. Mateescu was program committee member of FMICS-AVoCS'2017 (*International Workshop on Formal Methods for Industrial Critical Systems and Automated Verification of Critical Systems*), Torino, Italy, September 18–20, 2017.

[8] http://mcc.lip6.fr/models.php
[9] http://www.mars-workshop.org/

- G. Salaün was program committee member of PDP-4PAD'2017 (*25th Euromicro International Conference on Parallel, Distributed and Network-based Processing - Formal Approaches to Parallel and Distributed Systems*), St. Petersburg, Russia, March 6–8, 2017.
- G. Salaün was program committee member of SAC-SOAP'2017 (the *Service-Oriented Architectures and Programming* track) of SAC'2017 (*32nd ACM Symposium on Applied Computing*), Marrakesh, Morocco, April 3–7, 2017.
- G. Salaün was program committee member of SAC-SVT'2017 (the *Software Verification and Testing* track) of SAC'2017, Marrakesh, Morocco, April 3–7, 2017.
- G. Salaün and W. Serwe were program committee members of FSEN'2017 (*7th IPM International Conference on Fundamentals of Software Engineering*), Tehran, Iran, April 26–28, 2017.
- G. Salaün was program committee member of the poster track of ICSE'2017 (*39th International Conference on Software Engineering*), Buenos Aires, Argentina, May 20–28, 2017.
- G. Salaün was program committee member of COORDINATION'2017 (*19th International Conference on Coordination Models and Languages*), Neuchâtel, Switzerland, June 19–22, 2017.
- G. Salaün was program committee member of COMPSAC'2017 (*IEEE International Conference on Computers, Software, and Applications*), Torino, Italy, July 4–8, 2017.
- G. Salaün was program committee member of VBSP'2017 (*1st International Workshop on Verification of Business and Software Processes*), Paris, France, July 5, 2017.
- G. Salaün was program committee member of FACS'2017 (*14th International Conference on Formal Aspects of Component Software*), Braga, Portugal, October 10–13, 2017.
- G. Salaün was program committee member of Microservices'2017, Odense, Denmark, October 25–26, 2017.

9.1.2.3. Reviewer

- G. Barbon was a reviewer for COMPSAC'2017, SEFM'2017, and SAC-SVT'2018 (*33nd ACM Symposium on Applied Computing - Software Verification and Testing Track*), Pau, France, April 9–13, 2018.
- F. Lang was a reviewer for SEFM'2017 and FMICS-AVoCS'2017.
- L. Marsso was a reviewer for MARS'2017, COMPSAC'2017, SEFM'2017, and FMICS-AVoCS'2017.
- U. Ozeer was a reviewer for SEFM'2017.
- G. Salaün was a reviewer for MARS'2017.
- W. Serwe was a reviewer for MARS'2017, SEFM'2017, and ICTSS'2017 (*19th International Conference on Testing Software and Systems*), Paris, France, August 28–29, 2017.

9.1.3. Journal

9.1.3.1. Member of the Editorial Boards

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

9.1.3.2. Reviewer - Reviewing Activities

- F. Lang was a reviewer for STTT.
- R. Mateescu was a reviewer for STTT.
- W. Serwe was a reviewer for STTT, SPE (*Journal on Software: Practice and Experience*), and IJPEDS (*International Journal on Power Electronics and Drive Systems*).
- G. Salaün was a reviewer for JSC (*Journal of Symbolic Computation*), IEEE TSE (*Transactions of Software Engineering*), and SCP (*Science of Computer Programming*).

9.1.4. Software Dissemination and Internet Visibility

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

In 2017, the main facts are the following:

- We prepared and distributed twelve successive versions (2017-a to 2017-l) of CADP.
- We were requested to grant CADP licenses for 315 different computers in the world.

The CONVECS Web site [10] was updated with scientific contents, announcements, publications, etc.

H. Garavel started a major rewrite of CorTeX, a build system and a collection of tools for documents prepared using LaTeX.

By the end of December 2017, the CADP forum [11], opened in 2007 for discussions regarding the CADP toolbox, had over 414 registered users and over 1796 messages had been exchanged.

Also, for the 2017 edition of the Model Checking Contest, three families of models generated using CADP (totalling 64 Nested-Unit Petri Nets) were provided.

Other research teams took advantage of the software components provided by CADP (e.g., the BCG and OPEN/CAESAR environments) to build their own research software. We can mention the following developments:

- The COSTO tool for analyzing Kmelia components and services [48], [21]
- The VERCORS Platform for Model Checking Distributed Components [20]
- A Model-Driven and Multi-Agent Approach for Web Services Composition [19]
- Formal Analysis of Security Guidelines for Program Certification [57], [56], [58]
- A Product-Line for Families of Program Translators [22]
- The GROOVE Tool for Verification Based on Graph Rewriting [44]
- The FTRES Tool for Rare Event Simulation in Dynamic Fault Trees [52]
- The MIstRAL Tool for Middleware Reconfiguration Based on Formal Methods [51]
- The ALVIS Modelling Language for Embedded Systems [54]
- Adaptive Service Composition based on Runtime Verification of Formal Properties [23]

Other teams also used the CADP toolbox for various case studies:

- Assisting Refinement and Formal Verification in the Design of Embedded Systems [49]
- A Formal Model for Plastic Human Computer Interfaces [25]
- Verifying Concurrent Stacks by Divergence-Sensitive Bisimulation [55]
- Compositional Model Checking of Liveness Properties [59]
- Verification of Visibility-Based Properties on Multiple Moving Robots [53]

9.1.5. Invited Talks

- G. Barbon gave a talk entitled "*Debugging of Concurrent Systems using Counterexample Analysis*" on March 2nd, 2017 at the 2nd year PhD student day of the LIG.
- G. Barbon gave a talk entitled "*Debugging of Concurrent Systems using Counterexample Analysis*" on December 13, 2017 at the *Journée scientifique du pôle MSTIC*.
- H. Garavel gave two talks entitled "*Ten Different Ways on Defining Signed Integers Formally*" and "*Benchmarking Implementations of Conditional Term Rewrite Systems*" on February 28, 2017 at the Formal Methods seminar of Inria Grenoble – Rhône-Alpes.
- H. Garavel gave a talk entitled "*Nested-Units Petri Nets*" during OPCT'2017 (*Open Problems in Concurrency Theory*), a research seminar co-sponsored by the IFIP Working Group 1.8, that took place in the Institute of Science and Technology Austria (IST Austria), Vienna, on June 26–29, 2017.

[10] http://convecs.inria.fr
[11] http://cadp.inria.fr/forum.html

- H. Garavel gave a talk entitled "*From LOTOS to LNT*" during the ModelEd, TestEd, TrustEd Symposium in honour of the 60th birthday of Ed Brinksma held at the University of Twente, The Netherlands, on October 18, 2017.
- The members of CONVECS attended the 1st RIDINGS Workshop, held at Inria Grenoble – Rhône-Alpes on May 17, 2017. F. Lang gave a talk entitled "*The LNT language and the LNT2LOTOS compiler*". H. Garavel gave a talk entitled "*The Unheralded Value of the Multiway Rendezvous: Illustration with the Production Cell Benchmark*". L. Marsso gave a talk entitled "*A Large Term Rewrite System Modelling a Pioneering Cryptographic Algorithm*". G. Barbon gave a talk entitled "*Debugging of Concurrent Systems using Counterexample Analysis*".
- L. Marsso gave a talk entitled "*Formal Methods for Testing Networks of Controllers*" on May 29, 2017 at the 1st year PhD student day of the LIG.
- L. Marsso gave a talk entitled "*Formal Methods for Testing Networks of Controllers*" at the Scientific day of the ARC6 held at Université Grenoble Alpes, on November 23, 2017.
- L. Marsso and W. Serwe attended the 2nd RIDINGS Workshop held at Technical University Graz, Austria, on November 15, 2017. L. Marsso gave a talk entitled "*Testor: A Modular Tool for On-the-Fly Conformance Test Case Generation*". W. Serwe gave a talk entitled "*Using a Formal Model to Improve Verification of a Cache-Coherent System on Chip*".
- U. Ozeer gave a talk entitled "*Autonomous Resilience of Applications in a Largely Distributed Cloud Environment*" on May 29, 2017 at the 1st year PhD student day of the LIG.
- U. Ozeer gave a talk entitled "*Autonomous Resilience of Distributed IoT Applications in a Fog Environment*" at the IO Labs seminar held at Inria Paris on October 19–20, 2017.
- G. Salaün gave a talk entitled "*Checking Business Process Evolution*" on June 6, 2017 at the University of Málaga, Spain.

9.1.6. Research Administration

- H. Garavel was appointed to the Executive Commission in charge of International Relations at COMUE Université Grenoble Alpes.
- F. Lang is chair of the "*Commission du développement technologique*", which is in charge of selecting R&D projects for Inria Grenoble – Rhône-Alpes.
- R. Mateescu is the scientific correspondent of the European and International Partnerships for Inria Grenoble – Rhône-Alpes.
- R. Mateescu is a member of the "*Comité d'orientation scientifique*" for Inria Grenoble – Rhône-Alpes.
- R. Mateescu is a member of the "*Bureau*" of the LIG laboratory.
- G. Salaün is a member of the Scientific Committee of the PCS action of the PERSYVAL Labex.
- W. Serwe is (together with Laurent Lefèvre from the AVALON Inria project-team) correspondent in charge of the 2017 Inria activity reports at Inria Grenoble – Rhône-Alpes.
- W. Serwe is a member of the "*Comité de Centre*" at Inria Grenoble – Rhone-Alpes.
- W. Serwe is "*chargé de mission*" for the scientific axis *Formal Methods, Models, and Languages* of the LIG laboratory.

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

CONVECS is a host team for the computer science master entitled "*Mathématiques, Informatique, spécialité : Systèmes et Logiciels*", common to Grenoble INP and Université Grenoble Alpes (UGA).

In 2017, we carried out the following teaching activities:

G. Barbon gave a tutorial course on "*Language Theory*" (18 hours "*équivalent TD*" on formal languages, automata, regular expressions, and grammars) to second year students of ENSIMAG.

H. Garavel, together with Laurence Pierre (TIMA, Grenoble), created a new curriculum HECS [12] ("*High-confidence Embedded and Cyberphysical Systems*") for 2nd-year MOSIG (*Master of Science in Informatics at Grenoble*) students. This curriculum opened for the first time in September 2016.

F. Lang, R. Mateescu, G. Salaün, and W. Serwe gave lectures on models for concurrency, temporal logics, equivalences, formal languages and verification (36 hours "*équivalent TD*") as part of the MOSIG/MACS-2 course ("*Modeling and Analysis of Concurrent Systems*") led by G. Salaün.

G. Barbon and W. Serwe supervised each a group of six teams in the context of the "*projet Génie Logiciel*" (55 hours "*équivalent TD*", consisting in 16 hours of lectures, plus supervision and evaluation), ENSIMAG, January 2017.

F. Lang gave a lecture on "*Modélisation et Vérification des Systèmes Concurrents et Temps-Réel*" (27 hours "*équivalent TD*") to third year students of ENSIMAG.

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.

L. Marsso gave a course on "*Algorithms and Web Programming*" (44 hours "*équivalent TD*") at the department MMI of IUT1 (UGA).

G. Salaün taught about 200 hours of classes (algorithmics, Web development, object-oriented programming, iOS programming) at the department MMI of IUT1 (UGA). He is also headmaster of the "*Services Mobiles et Interface Nomade*" (SMIN) professional licence (3rd year of university) at IUT1/UGA.

9.2.2. Supervision

PhD in progress: G. Barbon, "*Debugging Concurrent Programs using Model Checking and Mining Techniques*", Université Grenoble Alpes, since October 2015, G. Salaün and V. Leroy

PhD in progress: L. Marsso, "*Formal Methods for Testing Networks of Controllers*", Université Grenoble Alpes, since October 2016, R. Mateescu, W. Serwe, I. Parissis, and Ch. Deleuze

PhD in progress: A. Muroor Nadumane, "*Softwarization of Everything: IoT Service Composition*", Université Grenoble Alpes, since October 2017, G. Salaün, R. Mateescu, L. Noirie, and M. Le Pallec

PhD in progress: U. Ozeer, "*Autonomous Resilience of Applications in a Largely Distributed Cloud Environment*", Université Grenoble Alpes, since November 2016, X. Etchevers, G. Salaün, F.-G. Ottogalli, and J.-M. Vincent

9.2.3. Juries

- R. Mateescu was reviewer of Alexandre Duret-Lutz's Habilitation thesis, entitled "*Contributions to LTL and Omega-Automata for Model Checking*", defended at EPITA (Paris, France) on February 10, 2017.
- R. Mateescu was reviewer of Zhengkui Zhang's PhD thesis, entitled "*Time and Cost Optimization of Cyber-Physical Systems by Distributed Reachability Analysis*", defended at the University of Aalborg (Denmark) on March 28, 2017.
- R. Mateescu was reviewer of Simon Busard's PhD thesis, entitled "*Symbolic Model Checking of Multi-Modal Logics: Uniform Strategies and Rich Explanations*", defended at Université Catholique de Louvain (Belgium) on May 10, 2017.
- G. Salaün was reviewer of Imen Boudhiba's PhD thesis, entitled "*A Model-Based Testing framework for IOSTS enriched with function calls*", defended at Centrale-Supélec (Paris, France) on March 2, 2017.

[12] http://hecs.imag.fr

- G. Salaün was PhD committee president of Hosein Nazarpour's PhD thesis, entitled "*Monitoring Multi-threaded and Distributed (Component-Based) Systems*", defended at Université Grenoble Alpes on June 26, 2017.
- G. Salaün was PhD committee president of Jean-François Weber's PhD thesis, entitled "*Guiding and Controlling the Reconfigurations of Component-based Systems*", defended at Université de Franche-Comté (Besançon, France) on October 5, 2017.

9.3. Popularization

H. Garavel participates to the program committee and organization committee of FMF (*Formal Methods Forum*) [13], a series of industrial conferences on formal methods set up by the competitivity clusters Aerospace Valley and Minalogic, with the support of Inria and many other partners. The 7th FMF conference, devoted to formal methods and cybersecurity, was held on January 31, 2017. The 8th FMF conference, devoted to formal methods and autonomous vehicles, was held on October 10, 2017. Both events gathered a large audience.

L. Marsso, R. Mateescu, and Olivier Clozeau (Innovista Sensors) participated to the "*Forum 5i*" held on June 1st, 2017 at Grenoble (World Trade Center), where they held a stand dedicated to the results of the Bluesky project [14].

R. Mateescu gave a lecture entitled "*Validation d'applications embarquées par des jumeaux numériques formels*" at the *Journée thématique Minalogic sur la modélisation des systèmes cyber-physiques* held in Grenoble on November 16, 2017.

10. Bibliography

Major publications by the team in recent years

[1] X. Etchevers, G. Salaün, F. Boyer, T. Coupaye, N. De Palma. *Reliable Self-deployment of Distributed Cloud Applications*, in "Software: Practice and Experience", 2017, vol. 47, n° 1, pp. 3-20 [*DOI :* 10.1002/spe.2400], https://hal.inria.fr/hal-01290465

[2] H. Evrard, 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", March 2017, vol. 88, 33 p. [*DOI :* 10.1016/j.jlamp.2016.09.002], https://hal.inria.fr/hal-01412911

[3] H. Garavel, F. Lang, R. Mateescu. *Compositional Verification of Asynchronous Concurrent Systems using CADP*, in "Acta Informatica", June 2015, vol. 52, n° 4, 56 p. [*DOI :* 10.1007/s00236-015-0226-1], https://hal.inria.fr/hal-01247507

[4] H. Garavel, F. Lang, R. Mateescu, W. Serwe. *CADP 2011: A Toolbox for the Construction and Analysis of Distributed Processes*, in "International Journal on Software Tools for Technology Transfer", 2013, vol. 15, n° 2, pp. 89-107 [*DOI :* 10.1007/s10009-012-0244-z], http://hal.inria.fr/hal-00715056

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

[13] http://projects.laas.fr/IFSE/FMF
[14] http://www.minalogic.com/en/project/bluesky

[6] R. MATEESCU, P. POIZAT, G. SALAÜN. *Adaptation of Service Protocols using Process Algebra and On-the-Fly Reduction Techniques*, in "IEEE Transactions on Software Engineering", 2012 [*DOI :* 10.1109/TSE.2011.62], http://hal.inria.fr/hal-00717252

[7] R. MATEESCU, W. SERWE. *Model Checking and Performance Evaluation with CADP Illustrated on Shared-Memory Mutual Exclusion Protocols*, in "Science of Computer Programming", February 2012 [*DOI :* 10.1016/J.SCICO.2012.01.003], http://hal.inria.fr/hal-00671321

Publications of the year

Articles in International Peer-Reviewed Journals

[8] R. ABID, G. SALAÜN, N. DE PALMA. *Asynchronous synthesis techniques for coordinating autonomic managers in the cloud*, in "Science of Computer Programming", October 2017, vol. 146, pp. 87 - 103 [*DOI :* 10.1016/J.SCICO.2017.05.005], https://hal.inria.fr/hal-01630717

[9] X. ETCHEVERS, G. SALAÜN, F. BOYER, T. COUPAYE, N. DE PALMA. *Reliable Self-deployment of Distributed Cloud Applications*, in "Software: Practice and Experience", 2017, vol. 47, n^{o} 1, pp. 3-20 [*DOI :* 10.1002/SPE.2400], https://hal.inria.fr/hal-01290465

[10] H. EVRARD, 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", March 2017, vol. 88, 33 p. [*DOI :* 10.1016/J.JLAMP.2016.09.002], https://hal.inria.fr/hal-01412911

International Conferences with Proceedings

[11] G. BARBON, V. LEROY, G. SALAÜN. *Debugging of Concurrent Systems using Counterexample Analysis*, in "Fundamentals of Software Engineering (FSEN)", Tehran, Iran, Lecture Notes in Computer Science, Springer Verlag, April 2017, vol. 10522, pp. 20-34, https://hal.inria.fr/hal-01533401

[12] F. DURÁN, G. SALAÜN. *Verifying Timed BPMN Processes Using Maude*, in "19th International Conference on Coordination Languages and Models (COORDINATION)", Neuchâtel, Switzerland, J.-M. JACQUET, M. MASSINK (editors), Coordination Models and Languages, Springer, June 2017, vol. LNCS-10319, pp. 219-236, Part 5: Verification [*DOI :* 10.1007/978-3-319-59746-1_12], https://hal.inria.fr/hal-01538104

[13] H. GARAVEL. *On the Most Suitable Axiomatization of Signed Integers*, in "23rd International Workshop on Algebraic Development Techniques (WADT'2016)", Gregynog, Wales, UK, United Kingdom, P. JAMES, M. ROGGENBACH (editors), Springer, September 2017, vol. Lecture Notes in Computer Science, n^{o} 10644, pp. 120-134, https://hal.inria.fr/hal-01667321

[14] H. GARAVEL, L. MARSSO. *A Large Term Rewrite System Modelling a Pioneering Cryptographic Algorithm*, in "2nd Workshop on Models for Formal Analysis of Real Systems", Uppsala, Sweden, April 2017, vol. 244, pp. 129 - 183 [*DOI :* 10.4204/EPTCS.244.6], https://hal.inria.fr/hal-01511859

[15] H. GARAVEL, W. SERWE. *The Unheralded Value of the Multiway Rendezvous: Illustration with the Production Cell Benchmark*, in "2nd Workshop on Models for Formal Analysis of Real Systems", Uppsala, Sweden, April 2017, vol. 244, pp. 230 - 270 [*DOI :* 10.4204/EPTCS.244.10], https://hal.inria.fr/hal-01511847

[16] A. Krishna, P. Poizat, G. Salaün. *VBPMN: Automated Verification of BPMN Processes*, in "13th International Conference on integrated Formal Methods (iFM 2017)", Turin, Italy, September 2017, https://hal.inria.fr/hal-01591665

[17] P. Lago, F. Lang, C. Roncancio, C. Jiménez-Guarín, R. Mateescu, N. Bonnefond. *The ContextAct@A4H real-life dataset of daily-living activities Activity recognition using model checking*, in "10th International and Interdisciplinary Conference - CONTEXT 2017", Paris, France, P. Brézillon, R. Turner, C. Penco (editors), Lecture Notes in Computer Science, Springer Verlag, June 2017, vol. 10257, pp. 175-188 [*DOI :* 10.1007/978-3-319-57837-8_14], https://hal.archives-ouvertes.fr/hal-01551418

Scientific Books (or Scientific Book chapters)

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

References in notes

[19] N. Adadi, M. Berrada, D. Chenouni, B. Bounabat. *Proposition of Web Services Composition Approach Basing of Model-Driven Approach and Multi-Agent Systems*, in "Computer Modelling and New Technologies", February 2017, vol. 21, n^{o} 3, pp. 12–19

[20] R. Ameur-Boulifa, L. Henrio, O. Kulankhina, E. Madelaine, A. Savu. *Behavioural Semantics for Asynchronous Components*, in "Journal of Logical and Algebraic Methods in Programming", 2017, vol. 89, pp. 1–40

[21] P. André, C. Attiogbé, J. Mottu. *Combining Techniques to Verify Service-based Components*, in "Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development (MODELSWARD'2017), Porto, Portugal", L. F. Pires, S. Hammoudi, B. Selic (editors), SciTePress, February 2017, pp. 645–656

[22] D. A. O. Camacho. *A Product-Line for Families of Program Translators: A Grammar-Based Approach*, Université Catholique de Louvain, Belgium, August 2010

[23] G. M. M. Campos, N. S. Rosa, L. F. Pires. *Adaptive Service Composition Based on Runtime Verification of Formal Properties*, in "Proceedings of the 50th Hawaii International Conference on System Sciences (HICSS'2017), Hilton Waikoloa Village, Hawaii, USA", AIS Electronic Library (AISeL), January 2017

[24] D. Champelovier, X. Clerc, H. Garavel, Y. Guerte, C. McKinty, V. Powazny, F. Lang, W. Serwe, G. Smeding. *Reference Manual of the LOTOS NT to LOTOS Translator (Version 5.7)*, November 2012, Inria/VASY, 153 pages

[25] A. Chebieb, Y. A. Ameur. *A Formal Model for Plastic Human Computer Interfaces*, in "Frontiers of Computer Science", March 2017

[26] E. M. Clarke, E. A. Emerson, A. P. Sistla. *Automatic Verification of Finite-State Concurrent Systems using Temporal Logic Specifications*, in "ACM Transactions on Programming Languages and Systems", April 1986, vol. 8, n^{o} 2, pp. 244–263

[27] N. COSTE, H. HERMANNS, E. LANTREIBECQ, W. SERWE. *Towards Performance Prediction of Compositional Models in Industrial GALS Designs*, in "Proceedings of the 21th International Conference on Computer Aided Verification CAV'2009 (Grenoble, France)", A. BOUAJJANI, O. MALER (editors), Lecture Notes in Computer Science, Springer Verlag, July 2009, vol. 5643, pp. 204–218, http://hal.inria.fr/inria-00381657

[28] R. DE NICOLA, F. W. VAANDRAGER. *Action versus State Based Logics for Transition Systems*, Lecture Notes in Computer Science, Springer Verlag, 1990, vol. 469, pp. 407–419

[29] H. GARAVEL. *Nested-Unit Petri Nets: A Structural Means to Increase Efficiency and Scalability of Verification on Elementary Nets*, in "Proceedings of the 36th International Conference on Application and Theory of Petri Nets and Concurrency (PETRI NETS'15), Brussels, Belgium", R. R. DEVILLERS, A. VALMARI (editors), Lecture Notes in Computer Science, Springer Verlag, June 2015, vol. 9115, pp. 179–199

[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)", S. T. VUONG (editor), North Holland, December 1989, pp. 147–162

[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)", Berlin, B. STEFFEN (editor), Lecture Notes in Computer Science, Springer Verlag, March 1998, vol. 1384, pp. 68–84, Full version available as Inria Research Report RR-3352

[32] H. GARAVEL, 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)", M. KIM, B. CHIN, S. KANG, D. LEE (editors), Kluwer Academic Publishers, August 2001, pp. 377–392, Full version available as Inria Research Report RR-4223

[33] H. GARAVEL, F. LANG, R. MATEESCU. *Compiler Construction using LOTOS NT*, in "Proceedings of the 11th International Conference on Compiler Construction CC 2002 (Grenoble, France)", N. HORSPOOL (editor), Lecture Notes in Computer Science, Springer Verlag, April 2002, vol. 2304, pp. 9–13

[34] H. GARAVEL, R. MATEESCU, 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)", Berlin, M. B. DWYER (editor), Lecture Notes in Computer Science, Springer Verlag, May 2001, vol. 2057, pp. 217–234, Revised version available as Inria Research Report RR-4341 (December 2001)

[35] H. GARAVEL, W. SERWE. *State Space Reduction for Process Algebra Specifications*, in "Theoretical Computer Science", February 2006, vol. 351, n^{o} 2, pp. 131–145

[36] H. GARAVEL, J. SIFAKIS. *Compilation and Verification of LOTOS Specifications*, in "Proceedings of the 10th International Symposium on Protocol Specification, Testing and Verification (Ottawa, Canada)", L. LOGRIPPO, R. L. PROBERT, H. URAL (editors), North Holland, June 1990, pp. 379–394

[37] J.-P. GROS. *A Unifying Framework for Comparing and Implementing Probabilistic Models*, Univ. Grenoble Alpes, Grenoble, June 2017

[38] H. HANSSON, B. JONSSON. *A Logic for Reasoning about Time and Reliability*, in "Formal Aspects of Computing", 1994, vol. 6, n^{o} 5, pp. 512–535

[39] A. HARTMANNS, H. HERMANNS. *In the Quantitative Automata Zoo*, in "Science of Computer Programming", 2015, vol. 112, pp. 3–23

[40] M. HENNESSY, R. MILNER. *Algebraic Laws for Nondeterminism and Concurrency*, in "Journal of the ACM", 1985, vol. 32, pp. 137–161

[41] H. HERMANNS. *Interactive Markov Chains and the Quest for Quantified Quality*, Lecture Notes in Computer Science, Springer Verlag, 2002, vol. 2428

[42] W. IMTIAZ. *Formal Modeling of Smart Home Automation*, Univ. Grenoble Alpes, Grenoble, September 2017

[43] J. F. JENSEN, K. G. LARSEN, J. SRBA, L. K. OESTERGAARD. *Efficient Model-Checking of Weighted CTL with Upper-Bound Constraints*, in "Springer International Journal on Software Tools for Technology Transfer (STTT)", 2016, vol. 18, n^{o} 4, pp. 409–426

[44] S. JUNGES, D. GUCK, J.-P. KATOEN, A. RENSINK, M. STOELINGA. *Fault Trees on a Diet: Automated Reduction by Graph Rewriting*, in "Formal Aspects of Computing", July 2017, vol. 29, n^{o} 4, pp. 651–703

[45] F. KORDON, H. GARAVEL, L. M. HILLAH, F. HULIN-HUBARD, B. BERTHOMIEU, G. CIARDO, M. COLANGE, S. DAL ZILIO, E. AMPARORE, T. LIEBKE, J. MEIJER, A. MINER, C. ROHR, J. SRBA, Y. THIERRY-MIEG, J. VAN DE POL, K. WOLF. *Complete Results for the 2017 Edition of the Model Checking Contest*, June 2017, http://mcc.lip6.fr/2017/results.php

[46] J. MAGEE, J. KRAMER. *Concurrency: State Models and Java Programs*, 2006, Wiley, April 2006

[47] R. MATEESCU, 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)", J. CUELLAR, T. MAIBAUM, K. SERE (editors), Lecture Notes in Computer Science, Springer Verlag, May 2008, vol. 5014, pp. 148–164

[48] M. E.-H. MESSABIHI. *Contribution à la spécification et à la vérification des logiciels à base de composants : enrichissement du langage de données de Kmelia et vérification de contrats*, Université de Nantes, France, July 2011

[49] H. MOKRANI. *Assistance au raffinement et à la vérification formels dans la conception des systèmes embarqués*, TÉLÉCOM ParisTech, June 2014

[50] J. PARREAUX. *Panorama des modèles et outils de vérification pour les systèmes probabilistes*, University Rennes I and ENS Rennes, July 2017, Prepared at Inria Grenoble

[51] N. ROSA. *Middleware Adaptation through Process Mining*, in "Proceedings of the 31st IEEE International Conference on Advanced Information Networking and Applications (AINA'2017), Taipei, Taiwan", L. BAROLLI, M. TAKIZAWA, T. ENOKIDO, H. HSU, C. LIN (editors), IEEE, March 2017, pp. 244–251

[52] E. RUIJTERS, D. REIJSBERGEN, P. DE BOER, M. STOELINGA. *Rare Event Simulation for Dynamic Fault Trees*, in "Proceedings of the 36th International Conference on Computer Safety, Reliability, and Security (SAFECOMP'2017), Trento, Italy", S. TONETTA, E. SCHOITSCH, F. BITSCH (editors), Lecture Notes in Computer Science, Springer Verlag, September 2017, vol. 10488, pp. 20–35

[53] A. N. SHESHKALANI, R. KHOSRAVI, M. MOHAMMADI. *Verification of Visibility-Based Properties on Multiple Moving Robots*, Y. GAO, S. FALLAH, Y. JIN, C. LEKAKOU (editors), Springer International Publishing, 2017, pp. 51–65

[54] M. SZPYRKA, G. J. NALEPA, K. KLUZA. *From Process Models to Concurrent Systems in Alvis Language*, in "Informatica", 2017, vol. 28, n^{o} 3, pp. 525–545

[55] X. YANG, J. KATOEN, H. LIN, H. WU. *Verifying Concurrent Stacks by Divergence-Sensitive Bisimulation*, in "CoRR", June 2017, vol. abs/1701.06104

[56] Z. ZHIOUA, Y. ROUDIER, R. AMEUR-BOULIFA. *Formal Specification and Verification of Security Guidelines*, in "Proceedings of the IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC'2017)", IEEE, January 2017, pp. 267–273

[57] Z. ZHIOUA, Y. ROUDIER, R. AMEUR-BOULIFA. *Formal Specification of Security Guidelines for Program Certification*, in "Proceedings of the 11th International Symposium on Theoretical Aspects of Software Engineering (TASE'2017), Nice, France", IEEE, September 2017

[58] Z. ZHIOUA, Y. ROUDIER, R. AMEUR-BOULIFA, T. KECHICHE, S. SHORT. *Tracking Dependent Information Flows*, in "Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP'2017), Porto, Portugal", P. MORI, S. FURNELL, O. CAMP (editors), SciTePress, February 2017, pp. 179–189

[59] S. DE PUTTER, A. WIJS. *Compositional Model Checking Is Lively*, in "Proceedings of the 14th International Conference on Formal Aspects of Component Software (FACS'17), Braga, Portugal", J. PROENÇA, M. LUMPE (editors), Lecture Notes in Computer Science, Springer Verlag, October 2017, vol. 10487, pp. 117–136