IN PARTNERSHIP WITH:
CNRS

Institut polytechnique de Grenoble

Université Joseph Fourier (Grenoble 1)

Activity Report 2011

Project-Team VASY

Validation of Systems

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

RESEARCH CENTER
Grenoble - Rhône-Alpes

THEME
Embedded and Real Time Systems

Table of contents

Project-Team VASY

Keywords: Formal Methods, Model-Checking, Programming Languages, Safety, Distributed System

VASY is an INRIA project team that is also a team of the LIG laboratory, a joint research unit of Centre National de Recherche Scientifique, Grenoble INP, Université Joseph Fourier, and Université Pierre Mendès-France. This is the last yearly activity report for VASY, as it completed its scientific activities on December 31, 2011 after having reached the 12-year age limit for INRIA project teams. As of January 2012, the VASY team members have launched a new research team named CONVECS (see http://convecs.inria.fr).

1. Members

Research Scientists

Hubert Garavel [Senior Researcher INRIA, Team Leader, part-time since May 1, 2011]
Frédéric Lang [Junior Researcher INRIA]
Radu Mateescu [Senior Researcher INRIA, HdR]
Wendelin Serwe [Junior Researcher INRIA]

Faculty Member

Gwen Salaün [Associate Professor, Grenoble INP]

Technical Staff

Iker Bellicot [until November 30, 2011]
Yann Genevois [until February 28, 2011]
Rémi Hérilier [until November 30, 2011]
Christine McKinty [part-time, until November 30, 2011]
Vincent Powazny [until June 28, 2011]
Damien Thivolle [until February 28, 2011]

Post-Doctoral Fellow

Matthias Güdemann [since October 17, 2011]

Administrative Assistant

Helen Pouchot

2. Overall Objectives

2.1. Overview

Created on January 1st, 2000, the VASY project focuses on formal methods for the design of reliable systems.

We are interested in any system (hardware, software, telecommunication) that exhibits *asynchronous concurrency*, i.e., any system whose behavior can be modelled as a set of parallel processes governed by interleaving semantics.

For the design of reliable systems, we advocate the use of formal description techniques together with software tools for simulation, rapid prototyping, verification, and test generation.

Among all existing verification approaches, we focus on *enumerative verification* (also known as *explicit state verification*) techniques. Although less general than theorem proving, these techniques enable an automatic, cost-efficient detection of design errors in complex systems.

Our research combines two main directions in formal methods, the *model-based* and the *language-based* approaches:

- Models provide mathematical representations for parallel programs and related verification problems. Examples of models are automata, networks of communicating automata, Petri nets, binary decision diagrams, boolean equation systems, etc. From a theoretical point of view, research on models seeks for general results, independently of any particular description language.
- In practice, models are often too elementary to describe complex systems directly (this would be tedious and error-prone). Higher level formalisms are needed for this task, as well as compilers that translate high level descriptions into models suitable for verification algorithms.

To verify complex systems, we believe that model issues and language issues should be mastered equally.

2.2. Highlights

In 2011, Hubert Garavel received the prestigious Humboldt Research Award granted by the Alexander von Humboldt foundation (Bonn, Germany).

3. Scientific Foundations

3.1. Models and Verification Techniques

By verification, we mean comparison — at some abstraction level — of a complex system against a set of *properties* characterizing the intended functioning of the system (for instance, deadlock freedom, mutual exclusion, fairness, etc.).

Most of the verification algorithms we develop are based on the *labeled transition systems* (or, simply, *automata* or *graphs*) model, which consists of a set of states, an initial state, and a transition relation between states. This model is often generated automatically from high-level descriptions of the system under study, then compared against the system properties using various decision procedures. Depending on the formalism used to express the properties, two approaches are possible:

- *Behavioral properties* express the intended functioning of the system in the form of automata (or higher level descriptions, which are then translated into automata). In this case, the natural approach to verification is *equivalence checking*, which consists in comparing the system model and its properties (both represented as automata) modulo some equivalence or preorder relation. We develop equivalence checking tools that compare and minimize automata modulo various equivalence and preorder relations; some of these tools also apply to stochastic and probabilistic models (such as Markov chains).
- *Logical properties* express the intended functioning of the system in the form of temporal logic formulas. In this case, the natural approach to verification is *model checking*, which consists in deciding whether or not the system model satisfies the logical properties. We develop model checking tools for a powerful form of temporal logic, the *modal μ-calculus*, which we extend with typed variables and expressions so as to express predicates over the data contained in the model. This extension (the practical usefulness of which has been highlighted in many examples) provides for properties that could not be expressed in the standard μ-calculus (for instance, the fact that the value of a given variable is always increasing along any execution path).

Although these techniques are efficient and automated, their main limitation is the *state explosion* problem, which occurs when models are too large to fit in computer memory. We provide software technologies (see § 5.1) for handling models in two complementary ways:

- Small models can be represented *explicitly*, by storing all their states and transitions in memory (*exhaustive* verification).
- Larger models are represented *implicitly*, by exploring only the model states and transitions needed for the verification (*on the fly* verification).

3.2. Languages and Compilation Techniques

Our research focuses on high level languages with *executable* and *formal* semantics. The former requirement stems from enumerative verification, which relies on the efficient execution of high-level descriptions. The latter requirement states that languages lacking formal semantics are not suitable for safety critical systems (as language ambiguities usually lead to interpretation divergences between designers and implementors). Moreover, enumerative techniques are not always sufficient to establish the correctness of an infinite system (they only deal with finite abstractions); one might need theorem proving techniques, which only apply to languages with formal semantics.

We are working on several languages with the above properties:

- LOTOS is an international standard for protocol description (ISO/IEC standard 8807:1989), which combines the concepts of process algebras (in particular CCS and CSP) and algebraic abstract data types. Thus, LOTOS can describe both asynchronous concurrent processes and complex data structures. We use LOTOS for various industrial case studies and we develop LOTOS compilers, which are part of the CADP toolbox (see § 5.1).
- We contributed to the definition of E-LOTOS (*Enhanced*-LOTOS, ISO/IEC standard 15437:2001), a deep revision of LOTOS, which tries to provide a greater expressiveness (for instance, by introducing quantitative time to describe systems with real-time constraints) together with a better user friendliness. Our contributions to E-LOTOS are available on the WEB (see http://vasy.inria.fr/elotos).
- We are also working on an E-LOTOS variant, named LOTOS NT (LOTOS *New Technology*) [12], [1], in which we can experiment with new ideas more freely than in the constrained framework of an international standard. Like E-LOTOS, LOTOS NT consists of three parts: a *data part*, which enables the description of data types and functions, a *process part*, which extends the LOTOS process algebra with new constructs such as exceptions and quantitative time, and *modules*, which provide for structure and genericity. The languages differ in that LOTOS NT combines imperative and functional features, and is also simpler than E-LOTOS in some respects (static typing, operator overloading, arrays), which should make it easier to implement. We are developing several tools for LOTOS NT: a prototype compiler named TRAIAN (see § 5.2), a translator from (a subset of) LOTOS NT to LOTOS (see § 6.2.2), and an intermediate semantic model named NTIF (*New Technology Intermediate Form*) [7].

3.3. Implementation and Experimentation

As far as possible, we validate our results by developing tools that we apply to complex (often industrial) case studies. Such a systematic confrontation of implementation and experimentation issues is central to our research.

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 § 6.3) illustrates the diversity of applications:

- *Hardware architectures:* asynchronous circuits, multiprocessor architectures, systems on chip, networks on chip, bus arbitration protocols, cache coherency protocols, hardware/software codesign;
- *Databases:* transaction protocols, distributed knowledge bases, stock management;

- *Consumer electronics:* home networking, video on-demand;
- *Security protocols:* authentication, electronic transactions, cryptographic key distribution;
- *Embedded systems:* smart-card applications, air traffic control, avionic systems;
- *Distributed systems:* virtual shared memory, distributed file systems, election algorithms, dynamic reconfiguration algorithms, fault tolerance algorithms, cloud computing;
- *Telecommunications:* high-speed networks, network management, mobile telephony, feature interaction detection;
- *Human-machine interaction:* graphical interfaces, biomedical data visualization;
- *Bioinformatics:* genetic regulatory networks, nutritional stress response, metabolic pathways.

5. Software

5.1. The CADP Toolbox

Participants: Iker Bellicot, Hubert Garavel [contact person], Yann Genevois, Rémi Hérilier, Frédéric Lang, Radu Mateescu, Christine McKinty, Wendelin Serwe, Damien Thivolle.

We maintain and enhance CADP (*Construction and Analysis of Distributed Processes* – formerly known as CÆSAR/ALDÉBARAN *Development Package*) [9], a toolbox for protocols and distributed systems engineering (see http://cadp.inria.fr). In this toolbox, we develop and maintain the following tools:

- CÆSAR.ADT [3] 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.
- CÆSAR [11] is a compiler that translates LOTOS processes into either C code (for rapid prototyping and testing purposes) or finite graphs (for verification purpose). 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/CÆSAR [4] 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/CÆSAR plays a central role in CADP by connecting language-oriented tools with model-oriented tools. OPEN/CÆSAR 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 tools have been developed within the OPEN/CÆSAR environment, among which:
  - BISIMULATOR, which checks bisimulation equivalences and preorders,
  - CUNCTATOR, which performs on-the-fly 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 regular alternation-free μ-calculus 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 allow 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_DRAW, which builds a two-dimensional view of a graph,
  - BCG_EDIT, which allows to modify interactively the graph layout produced by BCG_DRAW,
  - 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_MERGE, which gathers graph fragments obtained from distributed graph construction,
  - 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 to specify in XTL evaluation and diagnostic generation fixed point algorithms for usual temporal logics (such as HML [60], CTL [53], ACTL [55], etc.).

- The connection between explicit models (such as BCG graphs) and implicit models (explored on the fly) is ensured by OPEN/CÆSAR-compliant compilers, e.g.:
  - BCG_OPEN, for models represented as BCG graphs,
  - CÆSAR.OPEN, for models expressed as LOTOS descriptions,
  - EXP.OPEN, for models expressed as communicating automata,
  - FSP.OPEN, for models expressed as FSP [66] descriptions,
  - LNT.OPEN, for models expressed as LOTOS NT descriptions, and
  - SEQ.OPEN, for models represented as sets of execution trace.

The CADP toolbox also includes TGV (*Test Generation based on Verification*), developed by the VERIMAG laboratory (Grenoble) and the VERTECS project team at INRIA Rennes.

The CADP tools are well-integrated and can be accessed easily using either the EUCALYPTUS graphical interface or the SVL [6] 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.

5.2. The TRAIAN Compiler

Participants: Hubert Garavel [contact person], Frédéric Lang.

We develop a compiler named TRAIAN for translating descriptions written in the LOTOS NT language (see § 3.2) into C programs, which will be used for simulation, rapid prototyping, verification, and testing.

The current version of TRAIAN performs lexical analysis, syntactic analysis, abstract syntax tree construction, static semantics analysis, and C code generation for LOTOS NT types and functions.

Although this version of TRAIAN is still incomplete (it does not handle LOTOS NT processes), it already has useful applications in compiler construction [8]. The recent compilers developed by the VASY project team — including AAL, EVALUATOR 4.0 (see § 6.1.6), EXP.OPEN 2.0 (see § 6.1.4), LNT2LOTOS (see § 6.2.2), NTIF (see § 3.2), PIC2LNT (see § 6.2.3), and SVL (see § 6.1.4) — all contain a large amount of LOTOS NT code, which is then translated into C code by TRAIAN.

Our approach consists in using the SYNTAX tool (developed at INRIA Rocquencourt) for lexical and syntactic analysis together with LOTOS NT for semantical aspects, in particular the definition, construction, and traversal of abstract trees. Some involved parts of the compiler can also be written directly in C if necessary. The combined use of SYNTAX, LOTOS NT, and TRAIAN proves to be satisfactory, in terms of both the rapidity of development and the quality of the resulting compilers.

The TRAIAN compiler can be freely downloaded from the VASY WEB site (see http://vasy.inria.fr/traian).

6. New Results

6.1. Models and Verification Techniques

6.1.1. The BCG Format and Libraries

Participants: Hubert Garavel, Frédéric Lang, Wendelin Serwe.

BCG (*Binary-Coded Graphs*) is both a file format for the representation of explicit graphs and a collection of libraries and programs dealing with this format. Version 1.0 of the BCG format was recently replaced by version 1.1, which can exploit the capabilities of 64-bit addressing.

In 2011, we continued to enhance the BCG libraries as follows:

- We extended the BCG_READ application programming interface with three new primitives so as to increase symmetry with the OPEN/CÆSAR application programming interface (see § 6.1.2).
- We fixed a memory corruption problem occurring with very long label strings; this problem would cause random crashes of the DISTRIBUTOR tool (see § 6.1.5).

6.1.2. The OPEN/CÆSAR and CÆSAR_SOLVE Libraries

Participants: Iker Bellicot, Hubert Garavel, Yann Genevois, Frédéric Lang, Radu Mateescu, Wendelin Serwe.

OPEN/CÆSAR is an extensible, modular, language-independent software framework for exploring implicit graphs. This key component of CADP is used to build simulation, execution, verification, and test generation tools.

In 2011, a bug in the CÆSAR_TABLE library has been corrected, which would cause segmentation faults when certain primitives of the Application Programming Interface were invoked on a bounded table.

CÆSAR_SOLVE is a generic software library based on OPEN/CÆSAR for solving boolean equation systems of alternation depth 1 (i.e., without mutual recursion between minimal and maximal fixed point equations) on the fly. This library is at the core of several CADP verification tools, namely the equivalence checker BISIMULATOR, the minimization tool REDUCTOR, and the model checkers EVALUATOR 3.5 and 4.0. The resolution method is based on boolean graphs, which provide an intuitive representation of dependencies between boolean variables, and which are handled implicitly, in a way similar to the OPEN/CÆSAR interface [4].

In 2011, we improved the parallel resolution algorithm of CÆSAR_SOLVE (see § 6.1.5).

6.1.3. The EVALUATOR Tool

Participants: Iker Bellicot, Hubert Garavel, Yann Genevois, Radu Mateescu.

EVALUATOR is a model checker that evaluates a temporal logic property on a graph represented implicitly using the OPEN/CÆSAR environment. EVALUATOR works on the fly, meaning that only those parts of the implicit graph relevant to verification are explored. The model checking problem is reformulated in terms of solving a boolean equation system. A useful feature of EVALUATOR is the generation of diagnostics (examples and counterexamples) explaining why a formula is true or false.

In version 3.5 of EVALUATOR, properties are described in regular alternation-free μ-calculus, a logic built from boolean operators, possibility and necessity modalities containing regular expressions denoting transition sequences, and fixed point operators without mutual recursion between least and greatest fixed points. The input language of the tool also enables the user to define parameterized temporal operators and to group them into separate libraries.

In version 4.0 of EVALUATOR ($5,000$ lines of SYNTAX code, $40,500$ lines of LOTOS NT code, and $13,100$ lines of C code), properties are written in MCL (*Model Checking Language*) [18], an extension of the regular alternation-free μ-calculus of EVALUATOR 3.5 with data-handling and fairness operators. In particular, EVALUATOR 4.0 can handle modalities and fixed point operators with data parameters, regular expressions extended with counters, operators inspired from programming languages ("**if-then-else**", "**for**", etc.), and operators (of alternation depth two) allowing to characterize complex infinite sequences.

In 2011, we continued the extensive testing of EVALUATOR 3.5 and 4.0 using our test base of $10,000$ BCG graphs and $3,800$ MCL formulas. This revealed three errors in EVALUATOR 4.0, which have been corrected. We also brought the following enhancements to MCL and EVALUATOR 4.0:

- We extended the set of MCL operators of alternation depth two with parameterized versions of the infinite looping and saturation operators, which allow to succinctly encode the presence (respectively, the absence) of accepting cycles in generalized Büchi automata. The evaluation of these parameterized operators is translated into parameterized boolean equation systems, which are instantiated into plain boolean equation systems and solved on the fly using the algorithms A3 and A4 (extended with marked cycle detection) of the CÆSAR_SOLVE library. This evaluation procedure has a complexity linear in the size of the degeneralized Büchi automaton, which is represented by the boolean equation system obtained after instantiation.
- We enhanced MCL by adding a data type for manipulating sets of natural numbers. This data type, equipped with the classical set operations (union, intersection, difference, insertion, deletion, membership, etc.), enables a succinct specification of temporal properties referring to the past, such as the fact that a certain set of events (represented by natural numbers) occurred on the transition sequences leading from the initial state to the current state.
- We added a new option to EVALUATOR 4.0 for displaying a set of regular expressions that over-approximate the set of visible actions (transition labels in the LTS) satisfying the action predicates occurring in the MCL formula. This feature enables to improve the efficiency of verification by hiding the set of actions other than those produced from the MCL formula, minimizing the LTS modulo a weak equivalence relation compatible with the formula, and then verifying the MCL formula on

the minimized LTS. This may increase the efficiency of verification by one order of magnitude, as reported in [40].

EVALUATOR 4.0 was officially integrated in CADP in March 2011. MCL and EVALUATOR 4.0 were used successfully for analyzing mutual exclusion protocols (see § 6.3.1) and hardware architectures (see § 6.3.2).

6.1.4. Compositional Verification Tools

Participants: Frédéric Lang, Radu Mateescu.

The CADP toolbox contains various tools dedicated to compositional verification, among which EXP.OPEN 2.1, PROJECTOR 3.0, BCG_MIN 2.0, and SVL 2.2 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). PROJECTOR implements behavior abstraction [58], [64] by taking into account interface constraints. BCG_MIN minimizes 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.

In 2011, we corrected one bug in PROJECTOR, two bugs in EXP.OPEN, and four bugs in SVL. We also enhanced these tools as follows:

- The generalized parallel composition operator proposed in [13], including the support for "m among n" synchronization, has been added to SVL, leading to version 2.3 of SVL.
- Together with Pepijn Crouzen (Saarland University), we pursued our work on the so-called "smart reduction" techniques for compositional verification. An article about smart reduction was published in an international conference [32].

  We improved smart reduction for stochastic branching bisimulation so as to cut stochastic transitions (according to the "maximal progress" assumption) as early as possible in intermediate parallel compositions, thus yielding state space reductions. With this new optimization, EXP.OPEN detects when some action (usually, an output) offered by some process can synchronize with corresponding actions (usually, inputs) offered by the other processes in all their states; if so, all stochastic transitions in choice with this action can safely be cut in every intermediate composition. This situation occurs frequently with Input/Output Interactive Markov Chains.

Additionally, we studied an alternative compositional verification approach named *partial model checking* [47]. Given a temporal logic formula φ to be evaluated on a set S of concurrent processes, partial model checking consists in transforming φ into another equivalent formula φ' to be evaluated on a subset of S. Formula φ' is constructed incrementally by choosing one process P in S and incorporating into φ the behavioral information corresponding to P — an operation called *quotienting*. Simplifications must be applied at each step, so as to maintain formulas at a tractable size.

We developed a prototype implementation of this approach using the generic software components of CADP:

- We extended the definition of quotienting given by [47] to support all features of the input language of EXP.OPEN 2.1, which enables networks of labeled transition systems to be described using parallel composition operators borrowed from various process algebras such as CCS, CSP, and LOTOS, including also LOTOS NT parallel composition and "m among n" synchronisation operators [13].
- We gave an executable definition of quotienting in terms of a synchronous product between a graph representation (called *formula graph*) of the formula φ and the process P, thus enabling quotienting to be implemented efficiently in EXP.OPEN. We extended EVALUATOR 3.5 to automatically generate the formula graph corresponding to a temporal logic formula.
- We proposed and implemented efficient formula simplifications by combining reductions modulo bisimulations and partial formula evaluation computed using boolean equation systems.

We used this prototype implementation to verify 28 temporal logic properties on the TFTP avionics protocol [14]. For several of these properties, partial model checking uses hundreds of times less memory than on-the-fly model checking using EVALUATOR. This work led to a publication in an international conference [38].

6.1.5. Parallel and Distributed Verification Tools

Participants: Iker Bellicot, Hubert Garavel, Rémi Hérilier, Radu Mateescu.

The CADP toolbox contains several components designed to take advantage of distributed computing facilities (such as clusters of machines) to perform large-scale verifications, namely: CÆSAR_NETWORK, a network communication library used by the other tools, DISTRIBUTOR and BCG_MERGE, two companion tools [10] that perform reachability analysis using a distributed state space exploration algorithm, and BES_SOLVE [35], a tool that solves boolean equations systems using the various resolution algorithms provided by the CÆSAR_SOLVE library (see § 6.1.2), including a distributed on-the-fly resolution algorithm.

In 2011, we continued enhancing these tools, taking advantage of the valuable feedback provided by Eric Madelaine (INRIA Sophia Antipolis) who used CADP on the PACAGRID cluster. We brought the following improvements:

- We performed careful code reviews of the CÆSAR_NETWORK library, and corrected nine bugs. We equipped this library with logging primitives that proved to be helpful for debugging distributed algorithms programmed above the CÆSAR_NETWORK library.
- We fixed three bugs in DISTRIBUTOR and two bugs in BCG_MERGE.
- We pursued the intensive testing campaign undertaken in 2010 for BES_SOLVE, using up to 100 concurrent processes running on the PIPOL and GRID5000 platforms. We focused our efforts on the distributed resolution algorithm for boolean equation systems of CÆSAR_SOLVE/BES_SOLVE, which was tested extensively on examples of boolean equation systems represented explicitly as text files or generated randomly according to various parameters, the resolution results being cross-checked against the sequential resolution algorithms provided by CÆSAR_SOLVE/BES_SOLVE.

  Several bugs (affecting memory leaks, handling of early termination, diagnostic generation, collecting of statistical information about the resolution) were fixed. Changes were also carried out on the code to simplify its structure, increase modularity, and improve readability. The convergence of the distributed resolution algorithm was accelerated by backward propagation of constants as soon as they have been discovered.
- The EVALUATOR 4.0 tool was extended with a new prototype algorithm allowing the distributed verification of an MCL formula on a graph using several machines connected by a network. This functionality was implemented by connecting the tool to the distributed resolution algorithm for boolean equation systems and experimented out on clusters of machines.
- Finally, we added to CADP four new tools named PBG_CP, PBG_INFO, PBG_MV, and PBG_RM. These tools respectively enable to copy, query, move, and delete the PBG (Partitioned BCG Graph) collection of files generated and used by DISTRIBUTOR and BCG_MERGE.

6.1.6. Other Tool Developments

Participants: Hubert Garavel, Yann Genevois, Rémi Hérilier, Frédéric Lang, Radu Mateescu, Wendelin Serwe.

To support the usage of CADP in industry and academia, we pursued our efforts to master the software quality of CADP:

- We added support for Mac OS X 10.7 ("Lion") and enhanced the documentation for Mac OS X.
- We corrected one bug in the INSTALLATOR installation assistant, two bugs in the TST platform-checking command, and brought two bug fixes and one usability enhancement in the EUCALYPTUS graphical user-interface. We also provided a workaround for supporting recent versions of UBUNTU.

- We continued building a comprehensive validation framework, based on non-regression testing and semantical checking for the CADP tools. This framework allows functional testing of individual tools as well as integration testing for several CADP tools used together to perform complex verification scenarios on various computing platforms and using various compilers.

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

- the KMELIA tools for component-based systems [50], developed at the University of Nantes (France);
- the VERCORS tool for unifying architectural and behavioral specifications of distributed components [52], developed at INRIA Sophia-Antipolis;
- the DAMASCO (*Discovery, Adaptation and Monitoring of Context-Aware Services and Components*) framework for composition and adaptation based on model transformation [54], developed at the University of Málaga (Spain);
- the SCOOP tool for symbolic optimizations of probabilistic processes [62], [74], developed at RWTH Aachen (Germany);
- the SLCO (*Simple Language of Communicating Objects*) environment [49], developed at Eindhoven University of Technology (The Netherlands);
- the MOTOR tool for probabilistic analysis of embedded systems [68], developed at the Embedded Systems Institute (Eindhoven, The Netherlands);
- the ALVIS modeling language for design and verification of embedded systems [72], developed at the AGH University of Science and Technology (Krakow, Poland).

6.2. Languages and Compilation Techniques

6.2.1. Compilation of LOTOS

Participants: Hubert Garavel, Wendelin Serwe.

The CADP toolbox contains several tools dedicated to the LOTOS language, namely the CÆSAR.ADT compiler [3] for the data type part of LOTOS, the CÆSAR compiler [11] for the process part of LOTOS, and the CÆSAR.INDENT pretty-printer.

In 2011, in addition to fixing four bugs in the CÆSAR and CÆSAR.ADT compilers, we improved the LOTOS-dedicated tools of CADP as follows:

- We revised the predefined type libraries and C code generated by CÆSAR and CÆSAR.ADT to suppress warnings emitted by recent versions of GCC.
- We enhanced the format in which values of singleton and tuple types are displayed to end users.
- We modified the predefined libraries for natural (unsigned) and integer (signed) types so that users can now indicate the precise number of bits (between 1 and 64) to be used for the machine representation of these types, and can also determine the precise range (lower and upper bound) to be used for values of these types.
- We further modified these two libraries to enable (optional) overflow and underflow checking during arithmetic operations on the natural and integer types.

6.2.2. Compilation of LOTOS NT

Participants: Hubert Garavel, Frédéric Lang, Christine McKinty, Vincent Powazny, Wendelin Serwe.

Regarding the LOTOS NT language — a variant of E-LOTOS created by the VASY project team — we worked along two directions:

- We continued enhancing the TRAIAN compiler (see § 5.2), which generates C code from LOTOS NT data type and function definitions. TRAIAN is distributed on the Internet and used intensively within the VASY project team as a development tool for compiler construction [8].

  In 2011, TRAIAN was essentially in maintenance mode. We updated its documentation and added to its predefined library a conversion function that was missing.

- The LNT2LOTOS, LNT.OPEN, and LPP tools convert LOTOS NT code to LOTOS, thus allowing the use of CADP to verify LOTOS NT descriptions. These tools are officially part of CADP since 2010 and have been used successfully for many different systems (see § 6.3.1, § 6.3.2, § 6.3.5, § 6.3.5, and § 6.3.3).

  In 2011, we continued enhancing these tools, of which we delivered four new releases. In addition to 13 bug fixes, the following enhancements have been brought:

  – The LOTOS NT language was extended with range types (which are interval subtypes of character, integer, or natural types) and predicate types (which are subtypes of existing types, a boolean predicate being used to determine the domain of each subtype).

  – The LOTOS NT language was enriched with the concept of "module pragmas", which specify implementation constraints for predefined types such as naturals, integers, and strings. Also, the predefined operations "**first**" and "**last**" have been added for enumerated types.

  – The LNT2LOTOS translator was made semantically stricter by adding checks for overflow and underflow when doing natural and integer arithmetics, checking that range type bounds and array type bounds belong to the domain of admissible values for their parent types, and adding additional checks for type pragmas.

  – The LOTOS code generated by the LNT2LOTOS translator was optimized by handling equation premises (in the data part) and boolean guards (in the behavior part) that are always false or always true. Other optimizations have been added for process definitions whose bodies are empty or only contain a call to another process, for "**case**" statements that are followed by no instruction or only a "**null**" instruction, and for "**while**" loops with an empty body.

  – The speed of processing LOTOS NT specifications containing several modules has been made between two and three times faster.

  – The error and warning messages issued by the LOTOS NT tools have been enhanced.

  – The reference manual has been corrected, reorganized and comprehensively edited. Two new appendices have been added, one that lists all the predefined functions, and another one (20 pages) giving the formal semantics of LOTOS NT.

6.2.3. Source-Level Translations between Concurrent Languages

Participants: Hubert Garavel, Rémi Hérilier, Frédéric Lang, Radu Mateescu, Gwen Salaün, Wendelin Serwe, Damien Thivolle.

Although process algebras are, from a technical point of view, the best formalism to describe concurrent systems, they are not used as widely as they could be [2]. Besides the steep learning curve of process algebras, which is traditionally mentioned as the main reason for this situation, it seems also that the process algebra community scattered its efforts by developing too many languages, similar in concept but incompatible in practice. Even the advent of two international standards, such as LOTOS (in 1989) and E-LOTOS (in 2001), did not remedy this fragmentation. To address this problem, we started investigating source-level translators from various process algebras into LOTOS or LOTOS NT, so as to widen the applicability of the CADP tools.

In 2011, in addition to the LNT.OPEN tool suite (see § 6.2.2), we worked on the following translators:

- We continued our work on the FLAC tool, which translates a FIACRE program into a LOTOS program automatically, for verification using CADP. In 2011, 2 bugs reported by users of FLAC were corrected. Those corrections led to revisions 74 and 75 of the FLAC code, which is available on the development forge dedicated to FIACRE compilers [1]. We collected new examples of FIACRE code to enhance our test suite, which now comprises 79 examples.
- BPEL (*Business Process Execution Language*) [61] is a language inspired by the π-calculus [67] and standardized by the OASIS consortium (led by IBM and Microsoft) to describe the orchestrations of Web services. BPEL depends on other W3C standard XML-related languages: XML Schema for data types, XPATH for data expressions, and WSDL for declaring the interfaces (communication links and link functions) of a Web service.

  Following interest expressed by research teams at MIT and the Polytechnic University of Bucharest, we designed translation rules from BPEL to LOTOS NT in order to formally verify BPEL services with CADP. We began to develop an automated translator.

  In 2011, following a remark by Charles Pecheur (Université Catholique de Louvain, Belgium) who spotted an error in the translation of BPEL processes into LOTOS NT, we corrected the translation of exception handling so that it no longer interferes with the atomicity mechanism. The complete translation algorithm is given in Damien Thivolle's PhD thesis [23]. We pursued the implementation of our BPEL to LOTOS NT translator and finalized the translation of XML Schema types and WSDL definitions.
- We considered the π-calculus [67], a process algebra based on mobile communication. We proposed a general method for translating the finite control fragment of the π-calculus (obtained by forbidding recursive invocations of an agent through parallel composition operators) into LOTOS NT. The mobile communication is encoded using the data types of LOTOS NT, each channel name being represented as a value of an enumerated data type. The binary synchronization of π-calculus is enforced by associating a LOTOS NT gate to each parallel composition operator present in the π-calculus specification and by tagging each synchronization with the unique identifiers of the sender and receiver agents. The translation preserves the operational semantics by mapping each transition of a π-calculus agent to a single transition of the resulting LOTOS NT term.

  In 2011, we have extended the π-calculus with data-handling features, with the goal of widening its possible application domains. This was done by extending the language grammar and the translation to support typed variables and data expressions. As language for describing data, we chose LOTOS NT: indeed, the data types and functions used in the π-calculus specification can be described in LOTOS NT and directly incorporated to the LOTOS NT code produced by translation. This results in an applied π-calculus, such as the variant of the calculus proposed in [45] for the verification of security properties.

  The PIC2LNT translator was extended accordingly. It now consists of $2,100$ lines of SYNTAX code, $3,100$ lines of LOTOS NT code, and 500 lines of C code. The tool was tested on 234 examples of π-calculus specifications, including most of the examples provided in the Mobility Workbench distribution.

6.3. Case Studies and Practical Applications

6.3.1. Mutual Exclusion Protocols

Participants: Radu Mateescu, Wendelin Serwe.

[1] http://gforge.enseeiht.fr/projects/fiacre-compil

Mutual exclusion protocols are an essential building block of concurrent systems to ensure proper use of shared resources in the presence of concurrent accesses. Many variants of mutual exclusion protocols exist for shared memory, such as Peterson's or Dekker's well-known protocols. Although the functional correctness of these protocols has been studied extensively, relatively little attention has been paid to their performance aspects.

In 2011, we considered a set of 27 mutual exclusion protocols for up to sixteen processes with a shared memory and coherent local caches. We specified each protocol in LOTOS NT, using a set of generic modules to describe shared variables, the cache protocol, and the overall architectures (in total, $13,600$ lines of LOTOS NT code). Then, we compositionally added Markov delays modeling the latencies of read/write accesses on shared variables, so as to obtain the Interactive Markov Chain (IMC) corresponding to each protocol (up to 1.6 billion states and 2.7 billion transitions for the black-white bakery protocol [73] for four processes).

We verified functional properties using the same set of MCL [18] formulas for each protocol (in total, 380 lines of MCL). The mutual exclusion property was easy to express as an MCL formula, but other properties (livelock and starvation freedom, independent progress, and unbounded overtaking) turned out to be quite involved because they belong to the μ-calculus fragment of alternation depth two; fortunately, we succeeded in expressing them using the infinite looping operator of MCL, which can be checked in linear time. In particular, it was challenging to express these properties using a parameter N for the number of processes.

Finally, using the performance evaluation tools of CADP (i.e., BCG_STEADY for small numbers of processes and CUNCTATOR for larger numbers of processes), we computed the steady-state throughputs of critical section accesses by varying several parameters (relative speeds of processes, ratio between the time spent in critical and non-critical sections, etc.).

These experiments enabled us to compare the protocols according to their efficiency (steady-state throughputs) and study also their scalability for an increasing number of processors. We observed that symmetric protocols are more robust when the difference in execution speed between processes is large, which confirms the importance of the symmetry requirement originally formulated by Dijkstra [56]. The quantitative results corroborated those of functional verification: the presence of (asymmetric) starvation of processes, detected using temporal formulas, was clearly reflected in their steady-state throughputs. Our results also corroborate experimental measures found in the literature [48].

6.3.2. The Platform 2012 Architecture

Participant: Wendelin Serwe.

In the context of the MULTIVAL contract (see § 7.1), STMICROELECTRONICS studied PLATFORM 2012, a many-core programmable accelerator for ultra-efficient embedded computing in nanometer technology. This flexible and configurable multi-cluster platform fabric targets a range of emerging video, imaging, and next-generation immersive multimodal applications. Configurability options include the number of clusters, the number and type of processing elements (PE) per cluster, specialization of the architecture and instruction-set of the PEs, and finally, support of hardware-accelerated PEs. The platform is supported by a rich programming environment which embodies a range of platform programming models.

In 2011 we focused on the DTD (Dynamic Task Dispatcher) hardware block that assigns a set of application tasks on a set of PEs. It is called dynamic because each task itself might add tasks to the set of those to be dispatched by the DTD. The DTD is synthesized from a C++ model, optimized to generate an efficient hardware block. Due to the intrinsic complexity of the DTD, STMICROELECTRONICS was interested in the co-simulation of this C++ code with a formal model of the DTD.

In a first step, we generalized the LOTOS NT model of the DTD developed in 2010 to allow the handling of an arbitrary number of PEs ($1,200$ lines of LOTOS NT). We also modeled as LOTOS NT processes the different sets of tasks corresponding to various applications. To express the operations provided by the DTD, we had to include a call-stack in the model of each PE, as a means of circumventing the static-control constraints of CÆSAR forbidding recursion over parallel composition. STMICROELECTRONICS judged LOTOS NT to be essential in modeling the DTD, because using LOTOS instead would have been extremely difficult, requiring complex continuations with numerous parameters. We also wrote twelve scenarios ($1,000$ lines of LOTOS NT) describing applications to be dispatched by the DTD. For each scenario and for up to six PEs, we generated the corresponding LTS (up to 100 million states and 500 million transitions).

Then, for each generated LTS, we verified several properties, such as the correctness of assertions inserted in the model (by checking the set of transition labels), the termination of the scenario, or that each task is executed exactly once. We expressed the latter properties using the MCL language [18] and verified them using the EVALUATOR 4 model checker. This allowed us to point out a difference between our implementation and the one from the architect, highlighting a high sensibility on the order of terms in an equation, revealing an under-specified mechanism. We also verified the correctness of a complex optimization.

Having gained confidence in the LOTOS NT model, we applied the EXEC/CÆSAR framework to co-simulate the C++ and LOTOS NT models of the DTD, a challenge being the connection of the asynchronous LOTOS NT model with the synchronous C++ model, because one step of the C++ model corresponds, in general, to several transitions of the LOTOS NT model.

This case study enabled us to discover and correct a few bugs in CADP and led to a publication in an international conference [39].

6.3.3. The Self-configuration Protocol

Participant: Gwen Salaün.

Cloud computing emerged a few years ago as a major topic in modern programming. It leverages hosting platforms based on virtualization, and promises to deliver resources and applications that are faster and cheaper with a new software licensing and billing model based on the *pay-per-use* concept.

Distributed applications in the cloud are composed of a set of virtual machines (VMs) running a set of interconnected software components. However, the task of configuring distributed applications is complex as each VM includes many parameters either for local configuration (e.g., pool size, authentication data) or remote interconnection (e.g., IP address and port to access a server). Existing deployment solutions are often specific to certain applications and rarely take into account these configuration parameters, which are usually managed by dedicated scripts that do not work fully automatically.

Together with Xavier Etchevers, Thierry Coupaye (Orange labs), Fabienne Boyer, and Noël de Palma (INRIA Grenoble), we worked on the verification of an innovative self-configuration protocol [34] that automates the configuration of distributed applications in the cloud without requiring any centralized server nor a scripting effort. The high degree of parallelism involved in this protocol making its design difficult and error-prone, we decided to specify the protocol using LOTOS NT and to verify it with CADP. So doing, we detected a major bug, which was corrected in the reference JAVA implementation. The LOTOS NT specification also served as a workbench to experiment with several possible communication models, which helped us to avoid an erroneous design.

These results have been published in [44].

6.3.4. Realizability of Choreographies

Participants: Matthias Güdemann, Gwen Salaün.

The specification and the analysis of interactions among distributed components play an important role in service-oriented computing. In order to facilitate the integration of independently developed components (named *peers*) that may reside in different organizations, it is necessary to provide a global contract that the peers participating in a service composition should adhere to. Such a contract is called *choreography*. One important problem in a top-down development process is figuring out whether a choreography specification can be implemented by a set of peers that communicate via message passing. Given a choreography specification, it would be desirable to generate peers automatically by projecting the global choreography specification to each peer ignoring all messages that are not sent or received by that peer. However, generation of peers that precisely implement a choreography specification is not always possible, i.e., there are choreographies that are not implementable by a set of distributed peers. This problem is known as *realizability*.

In 2011, we considered the following aspects of the realizability problem:

- In collaboration with Gregor Gössler (INRIA Grenoble) we studied the realizability of choreographies for peers interacting asynchronously through message buffers. Although this problem is generally undecidable for unbounded buffers, we proposed techniques to check whether peers interacting asynchronously with finite buffers can realize a choreography, and if so, for which buffer sizes. These results have been published in [36].
- In collaboration with Pascal Poizat (LRI, Orsay), we proposed an approach to check the realizability of choreographies using the interaction model of BPMN (*Business Process Modeling Notation*) 2.0. While being a standard for the abstract specification of business workflows and collaboration between services, BPMN has only been recently extended into BPMN 2.0 to support choreographies. Our approach is based on a model transformation into LOTOS NT and the use of equivalence checking. We implemented a prototype of our approach using the ECLIPSE BPMN 2.0 editor and CADP. These results have been published in [43].
- In collaboration with Meriem Ouederni (LINA, Nantes), we studied the automatic synthesis of monitors to enforce realizability, using CADP to check equivalence between the choreography and an automatically obtained distributed implementation.

6.3.5. Other Case Studies

Participants: Hubert Garavel, Frédéric Lang, Radu Mateescu, Gwen Salaün, Wendelin Serwe, Damien Thivolle.

- In the context of the TOPCASED project (see § 7.2), we studied how CADP can be used to verify avionics protocols. In 2011, we prepared two lectures summarizing our prior results on four avionic protocols, namely a ground/plane communication protocol based on TFTP (*Trivial File Transfer Protocol*) [14], the BITE (*Built In Test Equipment*)/CMS (*Central Maintenance Function*), the ATC (*Air Traffic Control*) system, and the AFN (*Air Traffic System Facilities Notification*).
- Our prior work (2009–2010) with Fabienne Boyer and Olivier Gruber (Université Joseph Fourier Grenoble) on modeling and verification using LOTOS NT and CADP of the SYNERGY reconfiguration protocol led to a publication in an international conference [31].

Other teams also used the CADP toolbox for various case studies. To cite only recent work not already described in previous VASY activity reports, we can mention:

- behavior analysis of malware by rewriting-based abstraction [51];
- safety verification of fault-tolerant distributed components [46];
- verification of mobile ad hoc networks [57];
- model checking ERLANG applications [59];
- model-checking dataflow in service compositions [63];
- verification of a key chain based TTP transparent CEM protocol [65];
- semantics tuning of UML/SYSML [69];
- atomicity maintenance in EPCREPORT of ALE [70];
- cost analysis of semantic composability validation [71];
- rigorous development of prompting dialogs [75];
- scalably verifiable cache coherence [76].

7. Contracts and Grants with Industry

7.1. The Multival Project

Participants: Hubert Garavel, Rémi Hérilier, Frédéric Lang, Radu Mateescu, Christine McKinty, Vincent Powazny, Wendelin Serwe.

MULTIVAL (*Validation of Multiprocessor Multithreaded Architectures*) is a project of MINALOGIC, the *pôle de compétitivité mondial* dedicated to micro-nano technologies and embedded software for systems on chip (EMSOC cluster of MINALOGIC). It is funded by the French government (*Fonds Unique Interministériel*) and the *Conseil Général de l'Isère*. MULTIVAL addresses verification and performance evaluation issues for three innovative asynchronous architectures developed by BULL, CEA/LETI, and STMICROELECTRONICS.

MULTIVAL started in December 2006 and was extended until May 2011. In 2011, we focused our activities on the enhancement of CADP (see § 6.2.2 and § 6.2.3) and, in collaboration with our partners, on the verification of the DTD (see § 6.3.2).

7.2. The Topcased Project

Participants: Hubert Garavel, Frédéric Lang, Wendelin Serwe, Damien Thivolle.

TOPCASED (*Toolkit in OPen-source for Critical Application and SystEms Development*) is a project of AESE, the French *pôle de compétitivité mondial* dedicated to aeronautics, space, and embedded systems. This project gathers 23 partners, including companies developing safety-critical systems such as AIRBUS (leader), ASTRIUM, ATOS ORIGIN, CS, SIEMENS VDO, and THALES AEROSPACE.

TOPCASED develops a modular, open-source, generic CASE (*Computer-Aided Software Engineering*) environment providing methods and tools for embedded system development, ranging from system and architecture specifications to software and hardware implementation through equipment definition. VASY contributes to the combination of model-driven engineering and formal methods for asynchronous systems.

TOPCASED started in August 2006 and completed in December 2010. In 2011, we enhanced the FLAC translator from FIACRE to LOTOS (see § 6.2.3). We participated in the final review of TOPCASED and gave three lectures during the TOPCASED Days (see § 9.3).

During the International Paris Air Show (*salon international du Bourget*), the TOPCASED project received the AESE trophy of the best R&D project (category *Systèmes, équipements et logiciels pour l'aéronautique et l'espace*).

8. Partnerships and Cooperations

8.1. National Collaborations

Additionally, we collaborated in 2011 with the following INRIA project teams:

- OASIS (Sophia-Antipolis): distributed verification tools (Eric Madelaine);
- POP-ART (Rhône-Alpes): behavioral adaptation of software services and conformance checking of choreography specifications (Gregor Gössler);
- SARDES (Rhône-Alpes): verification of protocols for component-based architectures and virtualization (Fabienne Boyer, Olivier Gruber, and Noël de Palma).

Beyond INRIA, we had sustained scientific relations with the following researchers:

- Gaëlle Calvary and Sophie Dupuy (LIG, Grenoble);
- Pascal Poizat (LRI, Orsay);
- Meriem Ouederni (LINA, Nantes);
- Xavier Blanc and Cédric Teyton (LABRI, Bordeaux).

8.2. European Collaborations

The VASY project team is member of the FMICS (*Formal Methods for Industrial Critical Systems*) working group of ERCIM (see http://fmics.inria.fr). From July 1999 to July 2001, H. Garavel chaired this working group; since July 2002, he has been a member of the FMICS Board, and is in charge of dissemination actions. In November 2011, R. Mateescu was elected chairman of the FMICS working group.

In addition to our partners in aforementioned contractual collaborations, we had scientific relations in 2011 with several European universities and research centers, including:

- Polytechnic University of Bucharest (Valentin Cristea);
- Saarland University (Jonathan Bogdoll, Pepijn Crouzen, Arnd Hartmanns, and Holger Hermanns);
- University of Coimbra (Javier Camara);
- University of Málaga (Carlos Canal, Meriem Ouederni, and Ernesto Pimentel).

D. Thivolle defended his PHD thesis at the Polytechnic University of Bucharest on April 29, 2011.

Our long-term partnership with Saarland University has been strengthened by the Humboldt Forschungspreis received by H. Garavel, who started regular visits to Saarland University.

H. Garavel has participated in the review of the DFG (*Deutsche Forschungsgemeinschaft*) transregional project AVACS (*Automatic Verification And Analysis of Complex Systems*, see http://www.avacs.org) on September 14–15, 2011.

8.3. International Collaborations

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. Visits and Exchanges

In 2011, we had the following scientific exchanges:

- Nicolas Halbwachs (VERIMAG) visited us on January 28, 2011 and gave a talk entitled "*Analyse de programmes: propriétés numériques et tableaux*".
- Thomas Lambolais and Thanh-Liem Phan (Ecole des Mines d'Alès) visited us on February 9, 2011.
- Meriem Ouederni (University of Málaga, Spain) visited us from June 27 to July 1, 2011 and from November 21 to November 25, 2011.
- Freark van der Berg (University of Twente, The Netherlands) visited us on October 17–21, 2011.
- Farhad Arbab (CWI, Amsterdam, The Netherlands) visited us on November 22, 2011 and gave a talk entitled "*Interaction-Based Concurrency*".
- Gianluigi Zavattaro (University of Bologna, Italy) visited us on November 22, 2011 and gave a talk entitled "*Parameterized Verification of Ad Hoc Network Protocols* ".
- The annual VASY seminar was held in Autrans (France) on November 28–30, 2011.
- Xavier Blanc (LaBRI, Bordeaux) attended the VASY annual seminar and gave on November 28, 2011 a talk entitled "*Vpraxis et évolution d'applications Internet*".
- Christian Attiogbe (LINA, Nantes) attended the VASY annual seminar and gave on November 29, 2011 a talk entitled "*Composition dynamique de processus dans les systèmes complexes*".
- Grégory Batt (INRIA Rocquencourt) attended the VASY annual seminar and gave on November 30, 2011 a talk entitled "*A general computational method for robustness analysis with applications to synthetic gene networks*".
- Holger Hermanns (Saarland University) visited us on December 1st, 2011 and gave a LIG keynote entitled "*From Concurrency Models to Numbers: Performance, Dependability, Energy*".

9. Dissemination

9.1. Software Dissemination and Internet Visibility

The VASY project team distributes two main software tools: the CADP toolbox (see § 5.1) and the TRAIAN compiler (see § 5.2). In 2011, the main facts are the following:

- We prepared and distributed 11 successive beta-versions (from 2009-f to 2009-i and from 2010-a to 2010-g "Zurich") of CADP.
- The number of license contracts signed for CADP increased from 429 to 441.
- We were requested to grant CADP licenses for 631 different computers in the world.

We co-operated with the legal department to simplify the distribution procedure for academic users of CADP, who can now obtain the software using an entirely electronic procedure, which no longer requires the signature of paper copies of the license agreement.

We migrated all CADP documentation and Web site to reflect recent changes in email addresses (`cadp@inria.fr` instead of `cadp@inrialpes.fr`) and URLs (http://cadp.inria.fr instead of http://www.inrialpes.fr/vasy/cadp).

The VASY WEB site (see http://vasy.inria.fr) was updated with scientific contents, announcements, publications, etc. In 2011, we updated the CADP FAQ (list of frequently asked questions about CADP), aligning its structure with the CADP Forum.

In September 2007, we opened the "CADP Forum" (see http://cadp.inria.fr/forum.html) for discussions regarding the CADP toolbox. By the end of December 2011, this forum had over 195 registered users and over 1300 messages had been exchanged.

9.2. Program Committees

In 2011, the members of VASY took on the following responsibilities:

- H. Garavel was a program committee member for the TOPCASED *Days* conference, Toulouse, France, February 2–4, 2011.
- F. Lang was a program committee member for NEPTUNE'11 (*Nice Environment with a Process and Tools Using Norms and Example*), Paris, France, May 17–18, 2011.
- G. Salaün was publicity chair for DISCOTEC'11 (*6th International Federated Conferences on Distributed Computing Techniques*), Reykjavik, Iceland, June 6–9, 2011.
- H. Garavel was a program committee member for DCDS'11 (*3rd International Workshop on Dependable Control of Discrete Systems*), Saarbrücken, Germany, June 15–17, 2011.
- G. Salaün was a program committee member for CSSE'11 (*CSI International Symposium on Computer Science and Software Engineering*), Tehran, Iran, June 15–16, 2011.
- G. Salaün was a program committee member for AMMSE'11 (*2nd International Workshop on Algebraic Methods in Model-based Software Engineering*), Zürich, Switzerland, June 30, 2011.
- R. Mateescu was a program committee member for PDMC'11 (*10th International Workshop on Parallel and Distributed Methods in verifiCation*), Cliff Lodge, Snowbird, Utah, July 14, 2011.
- G. Salaün was co-chair of the program committee and co-editor of the proceedings for FMICS'11 (*16th International Workshop on Formal Methods for Industrial Critical Systems*), Trento, Italy, August 29-30, 2011. F. Lang was program committee member for FMICS'11.
- G. Salaün was a program committee member for VATSS'11 (*1st International Workshop on Verification, Analysis, and Testing of Service Systems*), Szeged, Hungary, September 4, 2011.

- G. Salaün was a program committee member for FOCLASA'11 (*10th International Workshop on the Foundations of Coordination Languages and Software Architectures*, Aachen, Germany, September 10, 2011.
- R. Mateescu was a program committee member for ECOWS'11 (*9th European Conference on Web Services*), Lugano, Switzerland, September 14–16, 2011.
- G. Salaün was a program committee member for FACS'11 (*8th International Symposium on Formal Aspects of Component Software*), Oslo, Norway, September 14–16, 2011.
- G. Salaün was a program committee member for QASBA'11 (*International Workshop on Quality Assurance for Service-Based Applications*), Lugano, Switzerland, September 14, 2011.
- G. Salaün was a program committee member for FLACOS'11 (*5th Workshop on Formal Languages and Analysis of Contract-Oriented Software*), Málaga, Spain, September 22–23, 2011.
- G. Salaün was a program committee member for VADER'11 (*2nd International Workshop on Variability, Adaptation and Dynamism in software systEms and seRvices*), Hersonissos, Crete, Greece, October 2011.
- G. Salaün is an editorial board member of the journal "*Service Oriented Computing and Applications*" published by Springer Verlag.

9.3. Lectures and Invited Conferences

In 2011, we gave talks in several international conferences and workshops (see bibliography below). Additionally:

- R. Mateescu and G. Salaün participated in the SARDES team seminar held at Allevard (France) on January 13, 2011. G. Salaün gave a talk entitled "*Translating Pi-calculus into* LOTOS NT".
- H. Garavel participated in the seminar of the 3rd-year license students of ENS-Lyon (*Ecole Normale Superieure de Lyon*) held in Le Pleynet (France) on January 26, 2011. He gave a talk entitled "*Modélisation et vérification de systèmes informatiques complexes*".
- H. Garavel, F. Lang, and D. Thivolle participated in the TOPCASED *Days* conference held in Toulouse (France) on February 2–4, 2011. F. Lang gave a talk entitled "*Formal Verification of Avionics Protocols using* CADP *in* TOPCASED WP3" on February 3, 2011. D. Thivolle gave a talk entitled "*Verification of GALS Systems using* TOPCASED *and* CADP *with an Application to a Ground/Plane Communication Protocol*" on February 4, 2011.
- H. Garavel gave a talk entitled "CADP *2010: A Toolbox for the Construction and Analysis of Distributed Processes*" at VERIMAG on March 3, 2011.
- G. Salaün gave a talk entitled "*Realizability of Choreographies using Process Algebra Encodings*" at LRI (Orsay, France) on March 28, 2011.
- G. Salaün gave a talk entitled "*Specifying and Verifying the* SYNERGY *Reconfiguration Protocol with* LOTOS NT *and* CADP" at the University of Málaga (Spain) on May 11, 2011.
- F. Lang participated in the ROCKS (*RigorOus dependability analysis using model ChecKing techniques for Stochastic systems*) held in Saarbrücken (Germany) on March 26–27, 2011. He gave a talk entitled "*Ten Years of Performance Evaluation of Concurrent Systems using* CADP".
- H. Garavel participated in the DCDS'11 (*3rd International Workshop on Dependable Control of Discrete Systems*) held in Saarbrücken (Germany) on June 15–17, 2011.
- H. Garavel participated in the 8th LASER Summer School on "Software Engineering – Tools for Practical Software Verification" held on Elba Island (Italy) on September 4–10, 2011. He gave a talk entitled "CADP *2010: A Toolbox for the Construction and Analysis of Distributed Processes*".
- R. Mateescu gave a keynote presentation entitled "*Model Checking and Performance Evaluation with* CADP *Illustrated on Shared-Memory Mutual Exclusion Protocols* at the SAFA'2012 workshop held in Sophia-Antipolis (France) on October 12, 2011.

- M. Güdemann participated in the seminar "*Journées scientifiques EA-ARC-ADT*" held in Paris (France) on November 16–17, 2011.
- H. Garavel and R. Mateescu participated in the FMICS working group seminar held in Paris on December 12th, 2011. H. Garavel gave a talk entitled "*Three Decades of Success Stories in Formal Methods*". R. Mateescu gave a talk entitled "*Analysis of Mutual Exclusion Protocols using* CADP".

9.4. Teaching Activities

The VASY project team is a host team for the computer science master entitled "*Mathématiques, Informatique, spécialité : Systèmes et Logiciels*", common to Grenoble INP and Université Joseph Fourier.

In 2011:

- H. Garavel, F. Lang, and W. Serwe gave, jointly with Pascal Raymond (CNRS, VERIMAG), a course on "*Méthodes formelles de développement*" to the computer science engineering students of CNAM (*Conservatoire National des Arts et Métiers*) Grenoble (30 hours).
- F. Lang and W. Serwe gave a course on "*Modélisation et Vérification des Systèmes Concurrents et Temps-Réel*" to the 3rd year students of ENSIMAG (18 hours).
- G. Salaün gave lectures on "*Algorithmics and Object-Oriented Programming*" to the 2nd year students of ENSIMAG (36 hours).
- G. Salaün gave a course on "*Specification, Verification, and Adaptation of Web Services*" to master students in Málaga, Spain, in May 2011 (10 hours).
- G. Salaün is co-responsible of the ISI (*Ingénierie des systèmes d'information*) department of ENSIMAG since September 1, 2011.
- R. Mateescu was a reviewer for Eric Madelaine's habilitation thesis, entitled "*Specification, Model Generation, and Verification of Distributed Applications*", defended at the University of Nice – Sophia-Antipolis on September 29, 2011.
- G. Salaün co-supervised the PhD thesis of Meriem Ouederni (University of Málaga, Spain), defended at the University of Málaga, Spain, on October 28, 2011.
- H. Garavel was a jury member for Matthias Raffelsiepers's PhD thesis, entitled "*Cell Libraries and Verification*", defended at the Technical University of Eindhoven on November 2, 2011.
- G. Salaün defended an habilitation thesis [22] on November 22, 2011.
- R. Mateescu was a reviewer for Rodrigo Tacla Saad's PhD thesis, entitled "*Parallel Model Checking for Multiprocessor Architecture*", defended at INSA Toulouse on December 20, 2011.

9.5. Miscellaneous Activities

Within the MINALOGIC *pôle de compétitivité mondial*, H. Garavel is a member of the operational committee of the EMSOC cluster (*Embedded System on Chip*).

H. Garavel is a member of the scientific council of the GIS (*Groupement d'Intérêt Scientifique*) consortium 3SGS on supervision, safety, and security of large systems.

H. Garavel is a member of the evaluation committee for the ANR (*Agence Nationale de la Recherche*) programme "*Ingénierie Numérique et Sécurité*".

H. Garavel is a member of the LIG working group in charge of restructuring the scientific themes of the laboratory.

H. Garavel participated in the selection committee for a *professeur des universités* position at the University of Nice – Sophia-Antipolis.

H. Garavel was a reviewer for the German Research Foundation (DFG), the Netherlands Organisation for Scientific Research (NWO), and the Dutch Technology Foundation (STW).

F. Lang is a member of the "*commission du développement technologique*", which is in charge of selecting R&D projects for INRIA Grenoble Rhône-Alpes.

R. Mateescu is the correspondent of the "*Département des Partenariats Européens*" for INRIA Grenoble Rhône-Alpes.

On November 1st, 2011, R. Mateescu was elected coordinator of the ERCIM Working Group FMICS (Formal Methods for Industrial Critical Systems) for the next three years.

G. Salaün is a member of the LIG Council (*Conseil de laboratoire*).

G. Salaün is a member of the ENSIMAG Council (*Conseil de l'école*).

10. Bibliography

Major publications by the team in recent years

[1] H. GARAVEL. *Défense et illustration des algèbres de processus*, in "Actes de l'Ecole d'été Temps Réel ETR 2003 (Toulouse, France)", Z. MAMMERI (editor), Institut de Recherche en Informatique de Toulouse, September 2003.

[2] H. GARAVEL. *Reflections on the Future of Concurrency Theory in General and Process Calculi in Particular*, in "Proceedings of the LIX Colloquium on Emerging Trends in Concurrency Theory (Ecole Polytechnique de Paris, France), November 13–15, 2006", C. PALAMIDESSI, F. D. VALENCIA (editors), Electronic Notes in Theoretical Computer Science, Elsevier Science Publishers, April 2008, vol. 209, p. 149–164, Also available as INRIA Research Report RR-6368, http://hal.inria.fr/inria-00191141.

[3] 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, p. 147–162.

[4] 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, p. 68–84, Full version available as Inria Research Report RR-3352, http://hal.inria.fr/inria-00073337.

[5] H. GARAVEL, H. HERMANNS. *On Combining Functional Verification and Performance Evaluation using CADP*, in "Proceedings of the 11th International Symposium of Formal Methods Europe FME'2002 (Copenhagen, Denmark)", L.-H. ERIKSSON, P. A. LINDSAY (editors), Lecture Notes in Computer Science, Springer Verlag, July 2002, vol. 2391, p. 410–429, Full version available as Inria Research Report 4492, http://hal.inria.fr/inria-00072096.

[6] 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, p. 377–392, Full version available as Inria Research Report RR-4223, http://hal.inria.fr/inria-00072396.

[7] H. GARAVEL, F. LANG. *NTIF: A General Symbolic Model for Communicating Sequential Processes with Data*, in "Proceedings of the 22nd IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems FORTE'2002 (Houston, Texas, USA)", D. PELED, M. VARDI (editors), Lecture Notes in Computer Science, Springer Verlag, November 2002, vol. 2529, p. 276–291, Full version available as Inria Research Report RR-4666, http://hal.inria.fr/inria-00071919.

[8] 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, p. 9–13.

[9] H. GARAVEL, F. LANG, R. MATEESCU, W. SERWE. *CADP 2006: A Toolbox for the Construction and Analysis of Distributed Processes*, in "Proceedings of the 19th International Conference on Computer Aided Verification CAV'2007 (Berlin, Germany)", W. DAMM, H. HERMANNS (editors), Lecture Notes in Computer Science, Springer Verlag, July 2007, vol. 4590, p. 158–163, http://hal.inria.fr/inria-00189021.

[10] H. GARAVEL, R. MATEESCU, I. SMARANDACHE. *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, p. 217–234, Revised version available as INRIA Research Report RR-4341 (December 2001).

[11] 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, p. 379–394.

[12] H. GARAVEL, M. SIGHIREANU. *Towards a Second Generation of Formal Description Techniques – Rationale for the Design of E-LOTOS*, in "Proceedings of the 3rd International Workshop on Formal Methods for Industrial Critical Systems FMICS'98 (Amsterdam, The Netherlands)", Amsterdam, J. F. GROOTE, B. LUTTIK, J. VAN WAMEL (editors), CWI, May 1998, p. 187–230, Invited talk.

[13] H. GARAVEL, M. SIGHIREANU. *A Graphical Parallel Composition Operator for Process Algebras*, in "Proceedings of the Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification FORTE/PSTV'99 (Beijing, China)", J. WU, Q. GAO, S. T. CHANSON (editors), Kluwer Academic Publishers, October 1999, p. 185–202.

[14] H. GARAVEL, D. THIVOLLE. *Verification of GALS Systems by Combining Synchronous Languages and Process Calculi*, in "Proceedings of the 16th International SPIN Workshop on Model Checking of Software SPIN'2009 (Grenoble, France)", C. PASAREANU (editor), Lecture Notes in Computer Science, Springer Verlag, June 2009, vol. 5578, p. 241–260, http://hal.inria.fr/inria-00388819.

[15] C. HELMSTETTER, O. PONSINI. *A Comparison of Two SystemC/TLM Semantics for Formal Verification*, in "Proceedings of the 6th ACM-IEEE International Conference on Formal Methods and Models for Codesign MEMOCODE'2008 (Anaheim, CA, USA)", IEEE Computer Society Press, June 2008, p. 59–68, http://hal.inria.fr/inria-00275456.

[16] R. MATEESCU, G. SALAÜN. *Translating Pi-Calculus into LOTOS NT*, in "Proceedings of the 8th International Conference on Integrated Formal Methods IFM'2010 (Nancy, France)", D. MERY, S. MERZ (editors), Lecture Notes in Computer Science, Springer Verlag, October 2010, vol. 6396, p. 229–244, http://hal.inria.fr/inria-00524586.

[17] R. MATEESCU, W. SERWE. *A Study of Shared-Memory Mutual Exclusion Protocols using CADP*, in "Proceedings of the 15th International Workshop on Formal Methods for Industrial Critical Systems FMICS'2010 (Antwerp, Belgium)", S. KOWALEWSKI, M. ROVERI (editors), Lecture Notes in Computer Science, Springer Verlag, September 2010, vol. 6371, p. 180–197, http://hal.inria.fr/inria-00532897.

[18] 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, n^o 5014, p. 148–164, http://hal.inria.fr/inria-00315312/fr/.

[19] O. PONSINI, W. SERWE. *A Schedulerless Semantics of TLM Models Written in SystemC via Translation into LOTOS*, 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, n^o 5014, p. 278–293, http://hal.inria.fr/inria-00259944.

[20] D. THIVOLLE, H. GARAVEL, X. CLERC. *Présentation du langage SAM d'Airbus*, INRIA/VASY, 16 pages, 2008, https://gforge.enseeiht.fr/docman/view.php/33/2745/SAM.pdf.

Publications of the year

Doctoral Dissertations and Habilitation Theses

[21] M. OUEDERNI. *On Checking the Compatibility of Service Interaction Protocols*, University of Málaga, October 2011.

[22] G. SALAÜN. *Process Calculi: Bridges and Application to Distributed Systems*, University of Grenoble, November 2011, Habilitation à Diriger des Recherches.

[23] D. THIVOLLE. *Langages modernes pour la vérification des systèmes asynchrones*, Université Joseph Fourier (Grenoble, France) and Universitatea Politehnica din Bucuresti (Bucharest, Romania), April 2011.

Articles in International Peer-Reviewed Journal

[24] C. CANAL, J. CÁMARA, G. SALAÜN. *Structural Reconfiguration of Components under Behavioral Adaptation*, in "Science of Computer Programming", 2012, To appear.

[25] F. DURÁN, M. OUEDERNI, G. SALAÜN. *A Generic Framework for N-Protocol Compatibility Checking*, in "Science of Computer Programming", 2012, To appear.

[26] R. MATEESCU, P. T. MONTEIRO, E. DUMAS, H. DE JONG. *CTRL: Extension of CTL with Regular Expressions and Fairness Operators to Verify Genetic Regulatory Networks*, in "Theoretical Computer Science", June 2011, vol. 412, n^o 26, p. 2854–2883, http://hal.inria.fr/inria-00610831/en.

[27] 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, To appear.

[28] R. MATEESCU, A. WIJS. *Sequential and Distributed On-the-Fly Computation of Weak Tau-Confluence*, in "Science of Computer Programming", 2012, To appear.

[29] N. ROOHI, G. SALAÜN. *Realizability and Dynamic Reconfiguration of Chor Specifications*, in "Informatica", 2011, http://hal.inria.fr/inria-00538959/en.

[30] G. SALAÜN, T. BULTAN. *Realizability of Choreographies using Process Algebra Encodings*, in "IEEE Transactions on Services Computing", 2012, To appear.

International Conferences with Proceedings

[31] F. BOYER, O. GRUBER, G. SALAÜN. *Specifying and Verifying the* SYNERGY *Reconfiguration Protocol with LOTOS NT/CADP*, in "Proceedings of the 17th International Symposium on Formal Methods FM'2011", Limerick, Ireland, M. BUTLER, W. SCHULTE (editors), Lecture Notes in Computer Science, Springer Verlag, June 2011, vol. 6664, p. 103–117, http://hal.inria.fr/hal-00648909/en.

[32] P. CROUZEN, F. LANG. *Smart Reduction*, in "Proceedings of Fundamental Approaches to Software Engineering FASE'2011", Saarbrücken, Germany, D. GIANNAKOPOULOU, F. OREJAS (editors), Lecture Notes in Computer Science, Springer Verlag, March 2011, vol. 6603, p. 111–126, http://hal.inria.fr/inria-00572535/en.

[33] C. EISENTRAUT, H. HERMANNS, L. ZHANG. *Concurrency and Composition in a Stochastic World*, in "Proceedings of the 21th International Conference on Concurrency Theory CONCUR 2010", Paris, France, P. GASTIN, F. LAROUSSINIE (editors), Lecture Notes in Computer Science, Springer Verlag, August 2011, vol. 6269, p. 21–39, http://hal.inria.fr/hal-00650728/en.

[34] X. ETCHEVERS, T. COUPAYE, F. BOYER, N. DE PALMA, G. SALAÜN. *Self-configuration of Legacy Distributed Applications in the Cloud*, in "4th IEEE/ACM International Conference on Utility and Cloud Computing", 2011, http://hal.inria.fr/hal-00665592/en.

[35] H. GARAVEL, F. LANG, R. MATEESCU, W. SERWE. *CADP 2010: A Toolbox for the Construction and Analysis of Distributed Processes*, in "Proceedings of the 17th International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS'2011", Saarbrücken, Germany, P. A. ABDULLA, K. RUSTAN M. LEINO (editors), Lecture Notes in Computer Science, Springer Verlag, March 2011, vol. 6605, p. 372–387, http://hal.inria.fr/inria-00583776/en.

[36] G. GÖSSLER, G. SALAÜN. *Realizability of Choreographies for Services Interacting Asynchronously*, in "Proc. of FACS'11", Lecture Notes in Computer Science, Springer, 2012, To appear.

[37] E. M. HAHN, H. HERMANNS, B. WACHTER, L. ZHANG. *PARAM: A Model Checker for Parametric Markov Models*, in "Proceedings of the 22nd International conference on Computer Aided Verification CAV 2010", Edinburgh, UK, T. TOUILI, B. COOK, P. JACKSON (editors), Lecture Notes in Computer Science, Springer Verlag, July 2011, vol. 6174, p. 660–664, http://hal.inria.fr/hal-00650737/en.

[38] F. LANG, R. MATEESCU. *Partial Model Checking using Networks of Labelled Transition Systems and Boolean Equation Systems*, in "Proceedings of the 18th International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS'2012", Talinn, Estonia, C. FLANAGAN, B. KÖNIG (editors), Lecture Notes in Computer Science, Springer Verlag, March 2012, To appear.

[39] E. LANTREIBECQ, W. SERWE. *Model Checking and Co-simulation of a Dynamic Task Dispatcher Circuit Using CADP*, in "Proceedings of the 16th International Workshop on Formal Methods for Industrial Critical Systems FMICS 2011", Trento, Italy, G. SALAÜN, B. SCHÄTZ (editors), Lecture Notes in Computer Science, Springer Verlag, August 2011, vol. 6959, p. 180–195, http://hal.inria.fr/hal-00642029/en.

[40] R. MATEESCU, A. WIJS. *Property-Dependent Reductions for the Modal Mu-Calculus*, in "Proceedings of the 18th International SPIN Workshop on Model Checking Software SPIN'2011", Snowbird, UT, USA, A. GROCE, M. MUSUVATHI (editors), Lecture Notes in Computer Science, Springer Verlag, July 2011, vol. 6823, p. 2–19, Full version available as INRIA Research Report RR-7690, http://hal.inria.fr/inria-00609585/en.

[41] M. OUEDERNI, G. SALAÜN, E. PIMENTEL. *Client Update: A Solution for Service Evolution*, in "8th International Conference on Services Computing (SCC'11)", Washington DC, USA, IEEE Computer Society Press, 2011, http://hal.inria.fr/hal-00649933/en.

[42] M. OUEDERNI, G. SALAÜN, E. PIMENTEL. *Measuring the Compatibility of Service Interaction Protocols*, in "26th ACM Symposium on Applied Computing", Taichung, Taiwan, China, ACM, 2011, http://hal.inria.fr/hal-00650529/en.

[43] P. POIZAT, G. SALAÜN. *Checking the Realizability of BPMN 2.0 Choreographies*, in "Proceedings of the 27th Symposium On Applied Computing SAC'12", Riva del Garda, Italy, ACM Press, 2012, To appear.

[44] G. SALAÜN, X. ETCHEVERS, N. DE PALMA, F. BOYER, T. COUPAYE. *Verification of a Self-configuration Protocol for Distributed Applications in the Cloud*, in "Proceedings of the 27th Symposium On Applied Computing SAC'12", Riva del Garda, Italy, ACM Press, 2012, To appear.

References in notes

[45] MARTÍN. ABADI, B. BLANCHET, C. FOURNET. *Just Fast Keying in the Pi Calculus*, in "ACM Transactions on Information and System Security", July 2007, vol. 10, n^{o} 3, p. 1–59.

[46] R. AMEUR-BOULIFA, R. HALALAI, L. HENRIO, E. MADELAINE. *Verifying Safety of Fault-Tolerant Distributed Components – Extended Version*, INRIA, September 2011, n^{o} RR-7717, http://hal.inria.fr/inria-00621264/en/.

[47] H. R. ANDERSEN. *Partial Model Checking*, in "Proceedings of the 10th Annual IEEE Symposium on Logic in Computer Science LICS (San Diego, California, USA)", IEEE Computer Society Press, June 1995, p. 398–407.

[48] T. E. ANDERSON. *The Performance of Spin Lock Alternatives for Shared Memory Multiprocessors*, in "IEEE Transactions on Parallel and Distributed Systems", January 1990, vol. 1, n^{o} 1, p. 6–16.

[49] S. ANDOVA, M. VAN DEN BRAND, L. ENGELEN. *Prototyping the Semantics of a DSL using ASF+SDF: Link to Formal Verification of DSL Models*, in "Proceedings of the Second International Workshop on Algebraic Methods in Model-based Software Engineering AMMSE (Zurich, Switzerland)", F. DURÁN, V. RUSU (editors), Electronic Proceedings in Theoretical Computer Science, Elsevier, June 2011, p. 65–79.

[50] P. ANDRÉ, G. ARDOUREL, C. ATTIOGBÉ. *Kmelia, un modèle abstrait et formel pour la description et la composition de composants et de services*, in "Technique et Science Informatiques", 2011, vol. 30, n^{o} 6, p. 627–658.

[51] P. BEAUCAMPS, I. GNAEDIG, J.-Y. MARION. *Behavior Analysis of Malware by Rewriting-based Abstraction - Extended Version*, May 2011, http://hal.inria.fr/inria-00594396.

[52] A. CANSADO, L. HENRIO, E. MADELAINE, P. VALENZUELA. *Unifying Architectural and Behavioural Specifications of Distributed Components*, in "Electronic Notes in Theoretical Computer Science", 2010, vol. 260, n^{o} 0, p. 25–45.

[53] 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, p. 244–263.

[54] J. CUBO, C. CANAL, E. PIMENTEL. *Context-Aware Composition and Adaptation based on Model Transformation*, in "Journal of Universal Computer Science", March 2011, vol. 17, n^{o} 5, p. 777–806.

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

[56] E. W. DIJKSTRA. *Solution of a Problem in Concurrent Programming Control*, in "Communications of the ACM", September 1965, vol. 8, n^{o} 9, p. 569–570.

[57] F. GHASSEMI, W. FOKKINK, A. MOVAGHAR. *Verification of mobile ad hoc networks: An algebraic approach*, in "Theoretical Computer Science", June 2011, vol. 412, n^{o} 28, p. 3262–3282.

[58] S. GRAF, B. STEFFEN, G. LÜTTGEN. *Compositional Minimization of Finite State Systems using Interface Specifications*, in "Formal Aspects of Computation", September 1996, vol. 8, n^{o} 5, p. 607–616.

[59] Q. GUO, J. DERRICK. *Formally based tool support for model checking Erlang applications*, in "International Journal on Software Tools for Technology Transfer (STTT)", August 2011, vol. 13, n^{o} 4, p. 355–376.

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

[61] D. JORDAN, J. EVDEMON. *Web Services Business Process Execution Language Version 2.0*, OASIS, Billerica, Massachussets, April 2007.

[62] J.-P. KATOEN, J. VAN DE POL, M. STOELINGA, M. TIMMER. *A linear process-algebraic format with data for probabilistic automata*, in "Theoretical Computer Science", January 2012, vol. 413, n^{o} 1, p. 36–57.

[63] N. KOKASH, C. KRAUSE, E. DE VINK. *Reo + mCRL2 : A framework for model-checking dataflow in service compositions*, in "Formal Aspects of Computing", August 2011.

[64] J.-P. KRIMM, L. MOUNIER. *Compositional State Space Generation from LOTOS Programs*, in "Proceedings of TACAS'97 Tools and Algorithms for the Construction and Analysis of Systems (University of Twente, Enschede, The Netherlands)", Berlin, E. BRINKSMA (editor), Lecture Notes in Computer Science, Springer Verlag, April 1997, vol. 1217, Extended version with proofs available as Research Report VERIMAG RR97-01.

[65] Z. LIU, J. PANG, C. ZHANG. *Verification of A Key Chain Based TTP Transparent CEM Protocol*, in "Electronic Notes in Theoretical Computer Science", 2011, vol. 274, p. 51–65.

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

[67] R. MILNER. *Communicating and Mobile Systems: the Pi-Calculus*, Cambridge University Press, 1999.

[68] B. NIELSON, K. LARSEN, J. TRETMANS. *Industrial Handbook*, Embedded Systems Institute (Eindhoven, The Netherlands), June 2011, n^{o} 512.

[69] I. OBER, I. OBER, I. DRAGOMIR, E. ABOUSSOROR. *UML/SysML semantic tunings*, in "Innovations in Systems and Software Engineering", August 2011, vol. 7, n^{o} 4, p. 257-264.

[70] J. SUN, H. ZHAO, W. WANG, G. HU. *Atomicity Maintenance in EPCreport of ALE*, in "Proceedings of the 10th IEEE/ACIS International Conference on Computer and Information Science ICIS 2011 (Sanya, Hainan Island, China)", IEEE Computer Society, May 2011, p. 224-229.

[71] C. SZABO, Y. M. TEO. *An Analysis of the Cost of Validating Semantic Composability*, in "Proceedings of the 25th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation PADS (Nice, France)", S. STRASSBURGER (editor), IEEE Computer Society, June 2011, p. 1–8.

[72] M. SZPYRKA, P. MATYASIK, R. MRÓWKA. *Alvis - Modelling Language for Concurrent Systems*, in "Intelligent Decision Systems in Large-Scale Distributed Environments", P. BOUVRY, H. GONZÁLEZ-VÉLEZ, J. KOLODZIEJ (editors), Springer Verlag, 2011, p. 315–341.

[73] G. TAUBENFELD. *The Black-White Bakery Algorithm and Related Bounded-Space, Adaptive, Local-Spinning and FIFO Algorithms*, in "Proceedings of the 18th International Conference on Distributed Computing DISC'04 (Amsterdam, The Netherlands)", R. GUERRAOUI (editor), Lecture Notes in Computer Science, Springer Verlag, October 2004, vol. 3274, p. 56–70.

[74] M. TIMMER. *SCOOP: A Tool for SymboliC Optimisations Of Probabilistic Processes*, in "Proceedings of the 8th International Conference on Quantitative Evaluation of SysTems QEST 2011 (Aachen, Germany)", C. PALAMIDESSI, A. RISKA (editors), IEEE Computer Society, September 2011, p. 315–341.

[75] K. J. TURNER, A. M. GILLESPIE, L. J. MCMICHAEL. *Rigorous Development of Prompting Dialogues*, in "Biomedical Informatics", September 2011, vol. 44, n^{o} 4, p. 713–727.

[76] M. ZHANG, A. R. LEBECK, D. J. SORIN. *Fractal Coherence: Scalably Verifiable Cache Coherence*, in "Proceedings of the 43rd Annual IEEE/ACM International Symposium on Microarchitecture MICRO 2010 (Atlanta, Georgia, USA)", IEEE Computer Society, December 2010, p. 471–482.