INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team VASY

Validation of Systems

Grenoble - Rhône-Alpes

THEME COM

Table of contents

1. Team

Head of Team
Hubert Garavel [DR2 INRIA]

Administrative Assistants
Stéphanie Berger [until January 31, 2007]
Marlyse Felici [from January 24 to August 31, 2007]
Diane Courtiol [since September 4, 2007]

Inria Staff
Frédéric Lang [CR1 INRIA]
Radu Mateescu [CR1 INRIA, HdR]
Wendelin Serwe [CR1 INRIA]

Invited Professor
Holger Hermanns [Saarland University and INRIA, since June 1st, 2007]

STMicroelectronics Staff
Etienne Lantreibecq

Software Engineers
David Champelovier
Xavier Clerc [since November 12, 2007]
Yves Guerte [since April 16, 2007]
Rémi Hérilier [since February 1st, 2007]
Romain Lacroix [since January 2nd, 2007]
Sylvain Robert [since February 1st, 2007]
Marie Vidal [until August 31, 2007]
Damien Thivolle [from September 1st to September 30, 2007]

Post-Doctoral Fellows
Claude Helmstetter [from July 23 to August 31, 2007]
Emilie Oudot [since September 1st, 2007]
Olivier Ponsini
Anton Wijs [since November 1st, 2007]

PhD Students
Nicolas Coste [STMICROELECTRONICS CIFRE grant, since March 1st, 2007]
Jan Stoecker [CORDI grant]
Damien Thivolle [MESR grant, since October 1st, 2007]
Meriem Zidouni [BULL CIFRE grant, since March 1st, 2007]

Visiting Scientists
David Sanán [University of Málaga, from March 1st to May 31, 2007]
Sandro Spina [University of Malta, from September 3 to October 24, 2007]

Student Interns
Nathalie Lépy [CNAM Grenoble, from February 19 to April 30, 2007]
Damien Thivolle [Université Joseph Fourier (Grenoble), from March 1st to July 31, 2007]

2. Overall Objectives

2.1. Introduction

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 comprises *asynchronous concurrency*, i.e., any system whose behavior can be modeled 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. 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 such a 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 such a case, the natural approach to verification is *model checking*, which consists in deciding whether the system model satisfies or not 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 was 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 § 4.1) for handling models in two complementary ways:

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

2.3. Languages and Compilation Techniques

Our research focuses on high level languages with an *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 a 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 a 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 § 4.1).
- 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://www.inrialpes.fr/vasy/elotos).
- We are also working on an E-LOTOS variant, named LOTOS NT (LOTOS *New Technology*) [11], [1], in which we can experiment 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 allows 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. Both 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 § 4.2), a translator from (a subset of) LOTOS NT to LOTOS (see § 5.2.2), and an intermediate semantic model named NTIF (*New Technology Intermediate Form*) [6].

2.4. Implementation and Experimentation

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

3. Application Domains

3.1. 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 virtually applicable to any system or protocol made of distributed agents communicating by asynchronous messages. The list of recent case studies performed with the CADP toolbox (see in particular § 5.3) 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;
- *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.

4. Software

4.1. The CADP Toolbox

Participants: David Champelovier, Hubert Garavel [contact person], Rémi Hérilier, Frédéric Lang, Radu Mateescu, Sylvain Robert, Wendelin Serwe, Damien Thivolle, Marie Vidal.

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

- CÆSAR.ADT [2] 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 [10] 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 [3] 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,
  - 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,
- 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 [64], CTL [50], ACTL [52], 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.:
  - CÆSAR.OPEN, for models expressed as LOTOS descriptions,
  - BCG_OPEN, for models represented as BCG graphs,
  - EXP.OPEN, for models expressed as communicating automata, and
  - SEQ.OPEN, for models represented as sets of execution traces.

The CADP toolbox also includes additional tools, such as ALDÉBARAN and TGV (*Test Generation based on Verification*) developed by the VERIMAG laboratory (Grenoble) and the VERTECS project-team of INRIA Rennes.

The CADP tools are well-integrated and can be accessed easily using either the EUCALYPTUS graphical interface or the SVL [5] scripting language. Both EUCALYPTUS and SVL provide users with an easy, 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.

4.2. The TRAIAN Compiler

Participants: David Champelovier, Hubert Garavel [contact person], Yves Guerte, Frédéric Lang.

We develop a compiler named TRAIAN for translating descriptions written in the LOTOS NT language (see § 2.3) 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 [7]. The recent compilers developed by the VASY project-team — namely AAL, CHP2LOTOS (see § 5.2.3), EVALUATOR 4.0, EXP.OPEN 2.0 (see § 5.1.4), FSP2LOTOS (see § 5.2.3), LNT2LOTOS (see § 5.2.2), NTIF (see § 2.3), and SVL (see § 5.1.4) — 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 traversals 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, as regards both the rapidity of development and the quality of resulting compilers.

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

5. New Results

5.1. Models and Verification Techniques

5.1.1. The CÆSAR_SOLVE Library

Participant: Radu Mateescu.

CÆSAR_SOLVE is a generic software library 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 model checker EVALUATOR 3.5 (see § 5.1.3), and the minimization tool REDUCTOR. 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 [3].

The CÆSAR_SOLVE library provides five different resolution algorithms. A1 and A2 are general algorithms based upon depth-first, respectively breadth-first, traversals of boolean graphs. A3 and A4, based upon memory-efficient depth-first traversals of boolean graphs, are optimized for the case of acyclic, respectively disjunctive/conjunctive, boolean graphs. A5 is a general algorithm based upon a depth-first traversal of boolean graphs; it generalizes Tarjan's algorithm for computing strongly connected components and is much faster than A1 and A2 when it is invoked many times on the same equation block. All these algorithms can generate diagnostics explaining why a result is true or false (examples and counterexamples).

In 2007, the CÆSAR_SOLVE library (13,600 lines of C code) was improved as follows:

- Two new resolution algorithms, named A6 and A7, based upon a breadth-first search of the dependency graph between boolean variables, were added to the library. Algorithm A6 (respectively, A7) is specialized for solving disjunctive minimal (respectively, conjunctive maximal) fixed point equation blocks when a single invocation of the algorithm is requested on the block, which is frequent in practice: equivalence checking leads to boolean equation systems containing a single conjunctive maximal fixed point equation block when one LTS is deterministic (for strong equivalence) and τ-free (for weak equivalences), therefore enabling the use of algorithm A7; model checking leads to systems containing a single disjunctive minimal fixed point equation block (for liveness properties expressing potential reachability) or a single conjunctive maximal fixed point equation block (for safety properties), both of which can be solved using algorithms A6 or A7, respectively.

  Algorithms A6 and A7 require less memory than the general algorithm A2 because they are optimized to store only boolean variables (and not the dependencies between them). When applied to equivalence checking and model checking problems, this amounts to store only the states of the LTSs involved, and not their transitions. Therefore, when applicable, algorithms A6 and A7 combine the advantages of algorithm A2 (small-depth diagnostics), and algorithms A3 and A4 (low memory consumption). The BISIMULATOR and EVALUATOR tools have been modified to select automatically algorithms A6 and A7 whenever appropriate. This leads in practice to memory savings of about 30% whilst keeping the same execution time.
- A new primitive was added for cleaning up the set of boolean variables explored during a sequence of resolution calls. Invoked periodically, this primitive achieves a trade-off between memory consumption and execution time for certain applications (such as reduction by τ-confluence) that make a large number of resolutions.
- The primitive for reading a boolean equation system from a text file was made reentrant, in such a way that applications using the library can read and manipulate several boolean equation systems at the same time.
- The primitive for displaying statistical information about the resolutions performed on a boolean equation system was enhanced to show global information about the whole boolean equation system, in addition to local information about individual equation blocks.

5.1.2. The BES_SOLVE Tool

Participants: Hubert Garavel, Radu Mateescu, Sylvain Robert.

To assess the CÆSAR_SOLVE library (see § 5.1.1), a prototype tool named BES_SOLVE was developed in the past years for generating random boolean equation systems and for reading/writing boolean equation systems from/to text files.

In 2007, we completely rewrote this tool, leading to BES_SOLVE 1.0, which became part of CADP in November 2007. The new version (3,500 lines of C code) allows to compare and cross-check the various resolution algorithms provided by the sequential and distributed versions of CÆSAR_SOLVE. It constructs a boolean equation system in memory, either by reading it from a (possibly compressed) text file, or by generating it randomly according to various parameters. Then, a boolean variable defined in some equation block of the boolean system can be solved by invoking any sequential or distributed algorithm of CÆSAR_SOLVE.

The random generation parameters are described in the newly defined RBC (*Random BES Configuration*) file format. An RBC file is a set of directives specifying the form of each random equation block of the boolean equation system. It allows to produce a diversity of boolean equation systems by varying 11 different parameters: fixed point sign of the current block, unique or multiple resolutions, algorithm used for solving variables, maximum number of variables, maximum number of operands in the boolean formulas occurring in the right-hand sides of equations, percentage of local (defined in the current block) and global (defined in other blocks) variables in the boolean formulas, percentage of disjunctive variables, percentage of false constants,

alternation between disjunctive and conjunctive variables between the left and right-hand sides of equations, and shape of the block (general, acyclic, disjunctive or conjunctive). Each parameter has a default value, which can be modified selectively for a subset of the equation blocks of the system.

A test suite of 110 boolean equation systems and 283 RBC files was collected and used to validate BES_SOLVE and CÆSAR_SOLVE across multiple platforms. It allowed to make random generation portable, so as to ensure that an RBC file yields the same random boolean equation system independently from the machine architecture.

5.1.3. The EVALUATOR 3.5 and 4.0 Tools

Participants: Hubert Garavel, Radu Mateescu, Sylvain Robert, Damien Thivolle.

EVALUATOR is a model checker that evaluates a temporal logic property on a graph represented implicitly using the OPEN/CÆSAR environment. 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 allows to define parameterized temporal operators and to group them into separate libraries. EVALUATOR works on the fly, meaning that only those parts of the implicit graph pertinent 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 2007, we enhanced the EVALUATOR 3.5 tool with a new option allowing to reduce the input graph on the fly modulo τ-confluence while checking the temporal formula. For systems containing many loosely-coupled parallel processes, this option can reduce execution time significantly (e.g., by a factor 7 on an Erathostene's sieve specified in LOTOS with 10 processes).

We also continued the development of the EVALUATOR 4.0 prototype tool (4,800 lines of SYNTAX code, 35,600 lines of LOTOS NT code, and 8,000 lines of C code), which accepts as input specifications written in MCL (*Model Checking Language*), an extension of the regular alternation-free μ-calculus of EVALUATOR 3.5 with data-handling operators. In addition to fixing several bugs and porting EVALUATOR 4.0 to new architectures and compilers, we enhanced the MCL language with a new "**check**" operator allowing to verify that a given state of the graph (either the initial state, or a state stored in the enclosing formula using the state capture operators) satisfies a certain state formula. This allows to express non-standard temporal properties, such as the fact that a state is nondeterministic w.r.t. a certain action. This work led to a publication in an international conference [24].

A comprehensive test suite containing $3,600$ formulas was collected and used to check upward compatibility between EVALUATOR 3.5 and EVALUATOR 4.0.

5.1.4. Compositional Verification Tools

Participants: Rémi Hérilier, Frédéric Lang, Sandro Spina.

The CADP toolbox contains various tools dedicated to compositional verification, among which EXP.OPEN 2.0, PROJECTOR 2.0, and SVL play a central role. EXP.OPEN 2.0 explores on the fly the graph corresponding to a network of communicating automata (represented as a set of BCG files). PROJECTOR 2.0 implements behavior abstraction [58], [67] by taking into account interface constraints. SVL (*Script Verification Language*) is both a high level language for expressing complex verification scenarios and a compiler dedicated to this language.

In 2007, we enhanced these tools along the following lines:

- We added to EXP.OPEN a new operator for specifying priorities between the transitions of communicating automata. This led to a new version 2.1 of EXP.OPEN that became part of CADP in January 2007.
- We entirely rewrote PROJECTOR by introducing an improved algorithm and more compact data structures. On average, the new version PROJECTOR 3.0 uses 3 times less memory than PROJECTOR 2.0 and runs 3 times faster (and even 36 times faster for some examples).

We experimented PROJECTOR 3.0 on the IEEE 1394 panic protocol (which is part of the IEEE 1394.1 "FIREWIRE" bridge specification). In 2005, PROJECTOR 2.0 could not generate the corresponding graph (due to memory exhaustion) on a PENTIUM III computer (700 MHz, 1 GB RAM, 4 GB swap memory) running LINUX. In 2007, using PROJECTOR 3.0, we managed to generate this graph (16 million states and 123 million transitions) in a few hours on the same machine.

- We enhanced the SVL language with new "**cut**" and "**stop**" operators. We also added means to pass C compiler options and BCG options to the CADP tools.

Together with Sandro Spina and Gordon Pace (University of Malta), we also started studying the combination of refined interface generation with graph reduction techniques to generate interfaces of tractable sizes.

5.1.5. Other Tool Developments

Participants: David Champelovier, Jérôme Fereyre, Hubert Garavel, Yves Guerte, Rémi Hérilier, Holger Hermanns, Romain Lacroix, Frédéric Lang, Nathalie Lépy, Radu Mateescu, Sylvain Robert, Wendelin Serwe, Marie Vidal.

Taking advantage of the new primitive added to the CÆSAR_SOLVE library (see § 5.1.1), we enhanced the on the fly τ-confluence reduction used by the REDUCTOR and DISTRIBUTOR tools so as to periodically clean the set of boolean variables explored while solving the underlying boolean equation system. This reduces the memory consumption of τ-confluence by up to 15%, whilst keeping the same execution speed and achieving the same reductions.

We developed and documented a new prototype tool named SCRUTATOR (600 lines of C code) that prunes on the fly a graph represented implicitly using the OPEN/CÆSAR environment. Two kinds of pruning are implemented by the tool: the first one deletes the states leading eventually to deadlocks, and the second one keeps only the states leading (potentially or eventually) to transitions labeled by a given action. Additionally, the tool can also reduce the pruned graph on the fly modulo τ-confluence, trace equivalence, and weak trace equivalence. To identify which states will be deleted or kept in the graph, SCRUTATOR builds a (disjunctive or conjunctive) boolean equation system, which is solved on the fly using algorithms A3 and A4 of the CÆSAR_SOLVE library. This allows to keep in memory only the states (and not the transitions) of the implicit graph during exploration. The use of SCRUTATOR to synthesize adaptors supervising the execution of distributed software components led to a publication [23].

Because of the growing usage of CADP in industry and academy, we set up an ambitious plan to master the software quality of CADP. To this aim, we started building a comprehensive validation framework, based on non-regression testing and semantical checking, for all the CADP tools. As mentioned already, we developed several test suites for boolean equation systems and μ-calculus formulas to validate BES_SOLVE (see § 5.1.2) and EVALUATOR (see § 5.1.3), respectively. We also collected a series of 1,700 BCG graphs that we used for the systematic testing of BCG_IO, BCG_MIN, BISIMULATOR, and PROJECTOR. This revealed one bug in BISIMULATOR and two bugs in BCG_MIN, which have been repaired, the latter in collaboration with Pepijn Crouzen (Saarland University, Germany).

We migrated 32-bit versions of CADP to support recent computing platforms (including SPARC stations running SOLARIS 10, INTEL machines running LINUX 2.6, and INTEL-based APPLE computers running MACOS 10.3 and 10.4) together with the latest C compilers available on these platforms (GCC 4, INTEL ICC 9.0, and SUN STUDIO 11).

A key objective for the perpetuity of CADP is the ability to support the latest 64-bit platforms. This is a heavy task because of the number of tools in CADP, of their intrinsic complexity, and their reliance upon third-party software. In the framework of the MULTIVAL project (see § 6.3), we made significant steps in this direction:

- We selected three 64-bit platforms (SPARC V9 stations running SOLARIS 10, INTEL IA64 machines running LINUX, and INTEL EMT64/AMD64 machines running LINUX) and we constructed cross-compiling environments for these platforms.
- In a first step towards 64-bit architectures, we improved the code of all CADP tools so as to suppress any warning message issued by recent C compilers with the strictest code-checking options.

- We ported all the 55 shell scripts of CADP to these 64-bit architectures.
- We started porting the BCG and OPEN/CÆSAR APIs to these 64-bit architectures.
- We started porting third party software used by CADP. The CUDD, TCL-TK, TIX, and Boehm-Demers garbage collector libraries have been ported successfully to 64-bit architectures. The SPARSE library has been ported partially. The FNC2 compiler generation system will not be ported, since it is no longer maintained and its source code not accessible to us.
- Together with Pierre Boullier, Philippe Deschamp, and Benoît Sagot (ATOLL/ALPAGE project-team of INRIA Rocquencourt), we started to port the SYNTAX compiler generation software (more than 300,000 lines of C code) to 64-bit architectures. We first removed all warning messages emitted by recent C compilers. We found and repaired several bugs, some of which were specific to 64-bit architectures. We set up automated procedures to compile SYNTAX on eleven different platforms (processor, operating system, and C compiler) as well as quality tests to be applied at each modification of SYNTAX. The port of SYNTAX is almost complete.

Finally, we designed a prototype IDE (*Integrated Development Environment*) for CADP within the ECLIPSE framework. In its current form (30,000 lines of JAVA code), it comprises both a LOTOS editor and a graphical user-interface (inspired from EUCALYPTUS, but based on ECLIPSE).

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:

- an abstract test case execution framework [79], developed at the University of Graz (Austria),
- an algorithm for interoperability test generation [53], developed at INRIA/University of Rennes,
- the C.OPEN and ANNOTATOR tools for model checking C programs [54], [55], [57], developed at the Universities of Málaga and Valencia (Spain),
- the LYS knowledge analysis toolset: an epistemic verification framework [81], developed at the Technical University of Eindhoven (The Netherlands).

5.2. Languages and Compilation Techniques

5.2.1. Compilation of LOTOS

Participants: Hubert Garavel, Wendelin Serwe.

The CADP toolbox contains several tools dedicated to the LOTOS language, namely the CÆSAR.ADT compiler [2] for the data type part of LOTOS, the CÆSAR compiler [10] for the process part of LOTOS, and the CÆSAR.INDENT pretty-printer.

In 2007, we performed maintenance activities for these tools, fixing a few errors. We also modified in many places the C code generated by CÆSAR and CÆSAR.ADT to avoid all warnings emitted by recent C compilers with the strictest code-checking options. For the end-user, this eases the development of LOTOS specifications by guaranteeing that all warnings seen when compiling the C code generated by CÆSAR and CÆSAR.ADT are relevant to the source LOTOS description (e.g., unused parameters of some LOTOS operation) or to the hand-written C code provided by the user (and are not meaningless warnings arising from C code generation artefacts).

We pursued our study of state space reduction techniques, our goal being to decrease the size of the graphs generated by CÆSAR, still preserving strong bisimulation between the original and reduced graphs. We improved the results of our reduction technique based on live variable analysis [9] by allowing data-flow optimizations to be repeatedly applied in a hierarchical manner on parts of the LOTOS specification. On 151 out of 684 benchmarks, we observed a reduction in the numbers of states (by a mean factor of 2.49 and a maximum factor of 24.27) and of transitions (by a mean factor of 2.41 and a maximum factor of 23.54).

5.2.2. Compilation of LOTOS NT

Participants: David Champelovier, Hubert Garavel, Yves Guerte, Frédéric Lang, Wendelin Serwe, Jan Stoecker.

As regards the LOTOS NT language — a variant of E-LOTOS elaborated by the VASY project-team — we worked in three directions:

- We continued to enhance our TRAIAN 2.5 compiler (see § 4.2), which generates C code from LOTOS NT data type and function definitions. TRAIAN is distributed on the Internet (see § 8.1) and used intensively within the VASY project-team as a development tool for compiler construction [7].

  In 2007, we corrected nine bugs (three bugs already known and six bugs discovered in 2007) and improved the C code generated by TRAIAN in two respects:

  - It no longer produces warnings when compiled with recent C compilers using the strictest code-checking options. Thus, TRAIAN users will only get warnings relevant to their LOTOS NT code.
  - It was ported to 64-bit architectures, together with the predefined includes and libraries provided with TRAIAN. We checked that this yields the same results as for 32-bit architectures.

  In addition, we fully automated the cross-compilation of TRAIAN for multiple platforms. We also enriched the validation database for TRAIAN with 60 new files, and enhanced the validation scripts to support cross-testing on multiple platforms.

  We undertook the development of a new version 3.0 of TRAIAN written entirely in LOTOS NT and no longer relying on the obsolete FNC2 compiler generation tool used for TRAIAN 1.* and 2.*. We completed the lexical and syntactic analysis phases of TRAIAN 3.0 representing 6,900 lines of code (4,500 lines of SYNTAX code, 2,000 lines of LOTOS NT code, and 400 lines of C code).

- In the context of the FORMALFAME PLUS contract (see § 6.2), we continued to improve the LNT2LOTOS tool suite that translates (a large subset of) LOTOS NT into LOTOS, thus allowing to use CADP to verify LOTOS NT descriptions. BULL is using this tool suite to model and verify critical parts of its FAME2 multiprocessor architecture (see § 5.3.1).

  In 2007, we brought the following improvements:

  - As regards the data part of LOTOS NT, a new type constructor "**array** $[n\,..\,m]$ **of** T" was added to LOTOS NT at BULL's request and implemented in the tool suite.
  - As regards the process part of LOTOS NT, we finalized the definition of processes and behaviors, ensuring language symmetry in the sense that most LOTOS NT constructs used in functions are also allowed in processes (absence of such a nice symmetry is a drawback of LOTOS making the language harder to learn). The concrete syntax and static semantics of processes have been specified. The translation of LOTOS NT process definitions is under development.

  LNT2LOTOS is developed using the SYNTAX/TRAIAN technology. Currently, it represents 22,300 lines of code (3,700 lines of SYNTAX code, 16,500 lines of LOTOS NT code, and 2,200 lines of C code). The LOTOS NT reference manual was updated and grew from 47 pages (at the end of 2006) to 61 pages.

- As regards the compilation of temporal extensions, we studied in 2007 the state of the art of intermediate forms for the compilation of high-level concurrent (and possibly timed) languages and undertook the definition of a new intermediate form into which languages combining data, concurrency, and real-time (such as E-LOTOS and LOTOS NT) could be translated efficiently. We took as a basis the NTIF model [6] for sequential processes with data, which we extended in several ways to support concurrency and real-time.

5.2.3. Source-Level Translations between Concurrent Languages

Participants: Xavier Clerc, Hubert Garavel, Rémi Hérilier, Frédéric Lang, Olivier Ponsini, 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 [21]. 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, so as to widen the applicability of the CADP tools.

In 2007, besides the LNT2LOTOS tool suite (see § 5.2.2), we worked on the following translators:

- In the framework of the FIACRE (see § 7.1), OPENEMBEDD (see § 6.4), and TOPCASED (see § 6.5) projects, and in cooperation with the LAAS-CNRS and IRIT laboratories, we continued the development of FIACRE (*Format Intermédiaire pour les Architectures de Composants Répartis Embarqués*). Derived from NTIF [6] and V-COTRE [47], FIACRE will be used as a pivot formalism between modeling languages (such as AADL, UML, or SYSML) and verification tools (such as CADP and TINA).

  In 2007, we finalized the formal definition of the syntax and semantics of FIACRE [31]. We started to develop support tools for FIACRE in close cooperation with LAAS-CNRS. A FIACRE to LOTOS translation algorithm covering the whole FIACRE language (except priorities and time constraints) was formalized [41]. We enriched the FIACRE compiler front-end developed by LAAS-CNRS with a verification module (550 lines of standard ML code) that checks whether all FIACRE variables are initialized before use. We started to develop a tool named FLAC (*Fiacre to Lotos Adaptation Component*) that translates FIACRE data types into LOTOS abstract data types (so far 3,050 lines of standard ML code).
- We considered CSP$_m$ (*Machine-readable* CSP) [77], a language implementing the CSP [75] process algebra designed at Oxford University (United Kingdom). A comparative study of CSP$_m$ and LOTOS led to the conclusion that translating CSP$_m$ to LOTOS would not be straightforward: even if CSP is close to LOTOS in essence (LOTOS having borrowed some CSP operators, and vice versa), this is not the case for CSP$_m$. Certain functional language features of CSP$_m$ (such as lambda expressions, lazy computation, and list comprehensions) proved to be difficult to translate to LOTOS because of the strict rewrite strategy enforced by the CÆSAR.ADT data type compiler. Moreover, the choice operator "[]" of CSP$_m$ does not translate easily to LOTOS. Our study [44] led to the development of a compiler front-end for CSP$_m$ (2,000 lines of SYNTAX code, 14,000 lines of LOTOS NT code, and 2,000 lines of C code), which currently covers syntax and static semantics (variable binding and type checking).
- We considered the process algebra FSP (*Finite State Processes*) defined in a popular textbook on concurrency [69] and supported by the LTSA tool designed at Imperial College (London, United Kingdom). We continued the development of the FSP2LOTOS translator, which translates FSP into LOTOS, EXP.OPEN, and SVL code. Our prior work on FSP2LOTOS led to a publication in an international conference [26].

  In 2007, we developed a new generic library (2,100 lines of C code), which allows to generate non-colliding LOTOS identifiers independently of the source language being translated. In particular, this solved a bug that occurred when the source FSP code contained identifiers identical to LOTOS keywords (e.g., "**i**", "**stop**", ...).

  The FSP2LOTOS translator was originally assumed to generate LOTOS/EXP.OPEN code that is branching equivalent to the source FSP code. In some cases, this assumption was wrong, due to the fact that branching equivalence is not a congruence for the priority operators of FSP. Therefore, we redesigned and reimplemented most FSP2LOTOS translation rules [42] so that the generated

LOTOS/EXP.OPEN code is now strongly equivalent (and thus, branching equivalent) to the source FSP code.

In order to validate the FSP2LOTOS tool experimentally, we developed a shell script that generates (using LTSA) a first graph corresponding to a source FSP specification, generates (using CADP) a second graph corresponding to the LOTOS/EXP.OPEN code generated by FSP2LOTOS, and checks (using the BISIMULATOR tool of CADP) that both graphs are equivalent once all labels in CADP format have been converted to the LTSA format. We used this script on a base of 574 FSP examples, including all the examples provided with the LTSA tool. Using the older version of FSP2LOTOS (December 2006), only 85% of the FSP examples would translate into branching equivalent LOTOS/EXP.OPEN code. Using the latest version of FSP2LOTOS (December 2007), strong equivalence held for all the 574 FSP examples.

- In the context of the INRIA/LETI collaboration (see § 7.1) and the MULTIVAL contract (see § 6.3), we continued the study of the process algebra CHP (*Communicating Hardware Processes*) for which the TIMA laboratory has developed a circuit synthesis tool named TAST [73] and which is used by the LETI laboratory to describe the FAUST2 architecture (see § 5.3.2). Our goal is to integrate formal verification tools into the design flow of complex asynchronous circuits.

  In 2007, we revisited the CHP2LOTOS translator to add support for CHP expressions containing multiple probes of channels (this feature provides for efficient hardware implementation by checking the availability of different partners in handshake communications). To guarantee an "atomic" evaluation of such expressions and ensure mutual exclusion between channels, we designed a locking mechanism that we implemented in the generated LOTOS code. Comparing the new version of the translator to the previous on 310 examples, we found that this increased the state space for 40 examples (by a maximum factor of two), but surprisingly also decreased the state space for 19 examples (by a maximum factor of two). We wrote an overview publication on the CHP to LOTOS translation that was submitted to an international journal.

- In the context of the INRIA/LETI collaboration (see § 7.1), we investigated the verification of TLM (*Transaction Level Model*) models. Compared to traditional RTL (*Register Transfer Level*) models, TLM allows faster simulation, simultaneous development of software and hardware, and earlier hardware/software partitioning. Among all languages supporting TLM, SYSTEMC [66] emerges as an industrial standard. SYSTEMC is a C++ library providing both a high-level description language and a simulation kernel that involves a central (largely deterministic) scheduler ordering the actions of the different processes.

  In 2007, we focused on the PV (*Programmers View*) level of SYSTEMC/TLM, i.e., TLM models without explicit timing information. We worked in two directions:

  – Our first approach follows the "official" simulation semantics of SYSTEMC/TLM, keeping the scheduler as defined in [66]. Inspired by a translation from SYSTEMC/TLM to PROMELA [78], we devised a translation from SYSTEMC/TLM to LOTOS, which we experimented on the parameterized benchmark with n concurrent components given in [78].

    Using the REDUCTOR and BISIMULATOR tools of CADP, we compared the state space generated for the LOTOS description with the set of finite execution traces produced by the RVS tool [62], [63] for SYSTEMC/TLM. Both were found to be trace equivalent, showing that our LOTOS-based approach complies to the SYSTEMC/TLM simulation semantics.

    Using the EVALUATOR tool of CADP (see § 5.1.3), we verified the liveness property given in [78] that explores the complete state space of the parameterized benchmark. We experimented both direct state space generation (for which we obtained results comparable to those reported in [78]) and compositional state space generation (for which we observed state spaces three times smaller than those reported for SPIN in [78], which enabled us to handle the case $n = 19$ while SPIN fails at $n > 15$).

– Our second approach pursues the work undertaken in 2006 to design a fully asynchronous semantics for SYSTEMC/TLM. In this approach, we get rid of the central scheduler because it implements a global supervisor (that is difficult to achieve in a parallel circuit without sacrificing efficiency) and hampers the effectiveness of formal verification by preventing certain possible execution sequences from being analyzed. To the contrary, our approach explores all possible interleavings arising from concurrent components and fits well with the principles of GALS (*Globally Asynchronous Locally Synchronous*), NOCs (*Networks on Chip*), and SOCs (*Systems on Chip*).

  We devised translation rules from SYSTEMC/TLM to LOTOS, which rely on the user to translate C++ types and functions to LOTOS manually and which are parameterized by a locking policy that controls the level of asynchrony between transactions. This work led to a publication in an international conference [25].

  We experimented our translation on several examples, namely the parameterized benchmark of [78] and seven examples taken from the SYSTEMC/TLM reference document [76], and we started investigating a large industrial case study provided by STMICROELECTRONICS.

We compared both approaches (i.e., with and without the scheduler) on the parameterized benchmark of [78]:

– We could show that the state space for the approach with scheduler is included (modulo branching preorder) in the state space for the approach without scheduler.

– We could show that the approach without scheduler generalizes the approach with scheduler in the sense that the latter is obtained from the former by choosing a particular locking policy. Remarkably, the approach without scheduler even allowed to handle the case $n = 19$ using either direct state space generation (102 million states and 152 million transitions) or compositional generation (3 million states and 23 million transitions, the largest intermediate state space having 91 million states and 128 million transitions).

– In practice, even if the approach without scheduler explores more configurations due to asynchrony, this is compensated in part by our efficient LOTOS encoding. For instance, the state spaces we generated for the approach without scheduler were similar in size to those reported in [78] with a scheduler-based semantics.

5.3. Case Studies and Practical Applications

5.3.1. The FAME2 Architecture

Participants: Hubert Garavel, Holger Hermanns, Radu Mateescu, Meriem Zidouni.

In the context of the MULTIVAL (see § 6.3) contract together with BULL, we studied the MPI software layer and MPI benchmark applications to be run on FAME2 (*Flexible Architecture for Multiple Environments*), a CC-NUMA multiprocessor architecture developed at BULL for teraflop mainframes and petaflop computing.

In 2007, we focused on a particular MPI benchmark (the so-called "*ping-pong*" protocol), our goal being to predict the performance of this benchmark on FAME2 machines, in particular to estimate the latency of send/receive operations in different topologies, different software implementations of the MPI primitives, and different cache coherency protocols. The ping-pong benchmark consists of two parallel processes, which send to each other a data packet k times. Two variants of the benchmark were specified using a combination of LOTOS code (to describe the behavior of processes) and C code (to describe the data structures of memory and caches). Several configurations were studied, for two data packets and k varying between 1 and 10 (which corresponds to a normal termination of the benchmark session) or being infinite (which corresponds to cyclic execution).

For each configuration, several safety and liveness temporal properties were specified in regular alternation-free μ-calculus, and successfully verified on the corresponding graph using EVALUATOR 3.5 (see § 5.1.3). Additionally, it was successfully checked using BISIMULATOR that the two variants were branching equivalent to each other (when observing only send/receive actions), and that each configuration with k iterations was smaller (modulo the branching preorder) than the configurations with $k+1$ iterations and k infinite.

Finally, the LOTOS specification was extended with Markov delays and a performance analysis was carried out using the BCG_MIN, DETERMINATOR, and BCG_STEADY tools in order to predict the latency of the send/receive operations and of the various types of transfers between memory and (local or remote) caches. The results obtained by numerical analysis were close enough to the experimental measures to assess the suitability of the model.

5.3.2. The FAUST Architecture

Participants: Frédéric Lang, Wendelin Serwe.

In the context of the INRIA/LETI collaboration (see § 7.1) and the MULTIVAL contract (see § 6.3), we pursued the study (started in 2005) of the FAUST (*Flexible Architecture of Unified System for Telecom*) NOC (*Network on Chip*) [46] developed by the LETI laboratory. Our previous work on the first version of FAUST led to a joint INRIA/LETI publication [27].

In 2007, we focused on the second version of FAUST, a NOC-based platform for wireless telecom applications (4G, MIMO, etc.). Together with LETI scientists, we studied the communication interconnect part of FAUST, which routes packets (consisting of several 34-bit flits) between the 23 components of the FAUST circuit. At the block level, this interconnect is described in the hardware process calculus CHP (*Communicating Hardware Processes*) and implemented, at the RTL level, in asynchronous logic. The interconnect has 23 communication nodes, each of which consisting of five input controllers and five output controllers. Each input controller dispatches incoming flits to one out of four output controllers, and each output controller arbitrates between four input controllers.

To generate the state space of the input controller, we first abstracted the CHP specification (500 lines of code) by applying data independence considerations: we replaced the 34-bit data values by 8-bit data values, focusing on the control information. Then, applying our CHP2LOTOS translator (see § 5.2.3) to the CHP description of the input controller, we produced a LOTOS specification in less than one second. This led to 15 concurrent LOTOS processes, several of which were so large that their parallel composition could not be computed directly. By replacing "internal" in CHP/LOTOS by "external" to reflect the actual hardware implementation of FAUST, we further reduced the size of the intermediate state spaces by a factor of 50, making the state space of the entire input controller tractable (less than 3 million states).

To generate the state space of the output controller, we also applied our CHP2LOTOS translator to the corresponding CHP description (400 lines of code). By making the same data abstractions as for the input controller, by applying symmetry reductions (allowing to consider only two out of four input controllers), and by using the compositional state space generation features of CADP, we managed to generate the state space for an output controller (less than 6 million states).

5.3.3. The xSTream Architecture

Participants: Nicolas Coste, Hubert Garavel, Holger Hermanns, Etienne Lantreibecq, Wendelin Serwe.

In the context of the MULTIVAL contract (see § 6.3) together with STMICROELECTRONICS, we studied XSTREAM, a multiprocessor dataflow architecture for high performance embedded multimedia streaming applications. In this architecture, computation nodes (e.g., filters) communicate by XSTREAM queues connected by a NOC (*Network on Chip*). An XSTREAM queue generalizes a bounded FIFO queue in two ways: it provides additional primitives (such as *peek* to consult items in the middle of the queue, which is not possible with the standard *push*/*pop* primitives of FIFO queues), and a *backlog* (extra memory) to allow the increase of the queue size when the queue overflows.

In 2007, we studied in depth the concurrent behavior of xSTREAM queues:

- We developed several LOTOS models of xSTREAM queues (in total 6800 lines of code), differing in whether each xSTREAM primitive (*push*, *pop*, *peek*, etc.) is modeled by one or two (request/response) LOTOS events.
- For several values of n and p, we established (by means of on the fly equivalence checking using the BISIMULATOR tool of CADP) that an xSTREAM queue of size n with a backlog of size p is branching equivalent to an xSTREAM queue of size $n + p$.
- We then investigated the interaction of several xSTREAM queues sharing the NOC. By using BISIMULATOR, we discovered that the state space (1.5 million states and 8 million transitions) for a pair (Q_1, Q_2) of xSTREAM queues sharing the NOC with another pair (Q_3, Q_4) of xSTREAM queues was not branching equivalent, when hiding the actions of (Q_3, Q_4), to the state space obtained for (Q_1, Q_2) alone. Such a possibility of starvation caused by NOC access conflicts revealed that the xSTREAM credit protocol was not optional (as assumed so far) but mandatory.
- We then incorporated the xSTREAM credit protocol into our LOTOS model and managed to generate the corresponding state space by means of compositional verification techniques (which allowed to reduce the size of the largest intermediate state space from 17 million states and 108 million transitions down to 1.5 million states and 8 million transitions). This time, we managed to prove branching equivalence, but only when the credit value parameter is equal to 1; for values greater than 1, we reported the issue to the xSTREAM architects, together with a convincing counterexample (a valid xSTREAM application running into a deadlock when executed on an xSTREAM simulator generated from our LOTOS model using the EXEC/CÆSAR environment of CADP).
- In parallel, we undertook performance evaluation studies to predict latency and throughputs in the communication architecture, as well as occupancy within xSTREAM queues (with and without backlogs). A key challenge is to combine probabilistic/stochastic information (e.g., the rates at which xSTREAM applications push and pop elements in and out of the queues) with precise timing information (e.g., memory access time). We investigated different techniques and tools, starting from the Interactive Markov Chain model supported by CADP tools (we used Erlang distributions to approximate precise timing information), but considering also max-plus algebra, discrete event simulation, and timed automata (using the UPPAAL/CORA tools).

5.3.4. Other Case Studies

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:

- the model-based testing of the registrar entity responsible for maintaining user location information within a session initiation protocol (such as voice over IP) [45],
- the verification of an on-line computer sale application based on Microsoft's .NET workflow foundation components [51],
- the formal analysis of Chaum's dining cryptographers protocol and the anonymity analysis of the Fujioka-Okamota-Ohta electronic voting scheme [81],
- the formal analysis of an implementation in C of Peterson's mutual exclusion protocol [56],
- the formal analysis of timed Erlang programs and OTP (*Open Telecom Platform*) library components [60], [59],
- the verification of "bridge exchanger" software connectors in deterministic dataflow applications [61],
- the verification of the identify phase of the IEEE 1394 "FIREWIRE" high performance serial bus [65],
- the verification of the RIES (*Rijnland Internet Election System*) voting protocol used in the Netherlands' Second Chamber election in 2006 [68],

- the verification of a cache coherence protocol for a JAVA DSM (*Distributed Shared Memory*) implementation [70],
- the formal analysis of the embedded software controlling the automatic document feeder of an OCÉ copier [71],
- the verification of automatically computed adaptors between evolving components of a library management application [72],
- the formal analysis of two typical wireless sensor network applications ("*Blink*" and "*Sense*") written in the NESC programming language for networked systems running TINYOS [74].

Finally, an overview paper of a collection of large case studies from a wide range of application areas [48] and a textbook [80], which uses CADP as a software support for teaching formal verification of systems and circuits, were published.

6. Contracts and Grants with Industry

6.1. The EC-MOAN Project

Participants: Hubert Garavel, Radu Mateescu, Emilie Oudot, Anton Wijs.

VASY participates to the EC-MOAN (*Scalable modeling and analysis techniques to study emergent cell behavior: Understanding the E. coli stress response*) project no. 043235, funded by the FP6 NEST-PATH-COM European program. It gathers seven participants: INRIA Rhône-Alpes (VASY and HELIX project-teams), Université Joseph Fourier (Grenoble), University of Twente, Free University of Amsterdam, University of Edinburgh, CWI Amsterdam, and Masaryk University Brno. EC-MOAN aims at the development of new, scalable methods for modeling and analyzing integrated genetic, metabolic, and signaling networks, and the application of these methods for a better understanding of a bacterial model system.

EC-MOAN started on February 1st, 2007 for three years. In 2007, our contributions focused on the definition of temporal specification formalisms allowing to express biologically-relevant properties (both branching-time properties, such as bistability, and linear-time properties, such as presence of oscillations) and their translation to the temporal specification formalisms accepted as input by CADP. We also started to devise new model checking algorithms for evaluating on the fly temporal operators on graphs resulting from qualitative simulation of genetic regulatory networks, such as those produced by the GNA (*Genetic Network Analyzer*) tool developed by the HELIX project-team.

6.2. The FormalFame Plus Contract

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

There is a long-standing collaboration between VASY and BULL, which aims at demonstrating that the formal methods and tools developed at INRIA can be successfully applied to BULL's multiprocessor architectures. The objective is to develop a complete and integrated solution supporting formal specification, simulation, rapid prototyping, verification, and testing.

Between 1995 and 1998, two case studies were successfully tackled using CADP: the POWERSCALE bus arbitration protocol [49] and the POLYKID multiprocessor architecture [13]. Between 1998 and 2004, the collaboration focused on FAME, the CC-NUMA multiprocessor architecture used in BULL's NOVASCALE series of high-performance servers based on INTEL ITANIUM processors. The CADP tools have been used to validate a crucial circuit of FAME – the FSS (*Fame Scalability Switch*) – that implements the cache coherency protocol.

In 2004, the collaboration was renewed by a followup contract named FORMALFAME PLUS, which, in 2005, was extended for two more years. FORMALFAME PLUS aims at enhancing the performance and usability of the CADP tools to address the FAME2 multiprocessor architecture (see § 5.3.1).

In 2007, VASY contributed to FORMALFAME PLUS by continuing the development of the LNT2LOTOS tool suite (see § 5.2.2), which allows to write architectural models in a higher level language than LOTOS. This tool suite was used by BULL to model and verify using CADP a critical part of its FAME2 architecture.

The FORMALFAME PLUS contract ended in March 2007 and found its continuation in the MULTIVAL project (see § 6.3).

6.3. The Multival Project

Participants: David Champelovier, Nicolas Coste, Hubert Garavel, Yves Guerte, Rémi Hérilier, Holger Hermanns, Romain Lacroix, Frédéric Lang, Etienne Lantreibecq, Radu Mateescu, Olivier Ponsini, Wendelin Serwe, Marie Vidal, Meriem Zidouni.

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 de compétitivité des entreprises*) and *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 for three years. In 2007, we focused our activities on the enhancement of CADP (see § 5.1 and § 5.2) and case studies in collaboration with our partners to verify and predict the performance of the architectures FAME2 (see § 5.3.1), FAUST2 (see § 5.3.2), and XSTREAM (see § 5.3.3).

6.4. The OpenEmbedd Project

Participants: Xavier Clerc, Hubert Garavel, Frédéric Lang, Radu Mateescu, Wendelin Serwe, Jan Stoecker.

OPENEMBEDD is a French national project of ANR (*Agence Nationale de la Recherche*), initiated by RNTL (*Réseau National des Technologies Logicielles*). The goal of OPENEMBEDD is to develop an open-source, generic, standard software engineering platform for real-time embedded systems, such as those developed by AIRBUS, CS, FRANCE TELECOM, and THALES. Within an ECLIPSE framework, this platform will combine the principles of model-driven engineering with those of formal methods.

OPENEMBEDD started in May 2006 for three years. In 2007, we contributed to requirement surveys on formal verification of embedded systems [28], [32], [34], we pursued the definition of the FIACRE asynchronous intermediate model for embedded systems [29] and studied the principles of an automated translation from FIACRE to LOTOS (see § 5.2.3).

6.5. The Topcased Project

Participants: Hubert Garavel, Frédéric Lang, Nathalie Lépy, Sylvain Robert, Jan Stoecker, 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 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 in the combination of model-driven engineering and formal methods for asynchronous systems.

TOPCASED started in August 2006 for 40 months. In 2007, we worked along the following lines:

- In collaboration with colleagues from LAAS-CNRS and IRIT (Toulouse, France), we reviewed existing intermediate models for asynchronous concurrent processes so as to select features suitable for TOPCASED [35]. This led to the definition of FIACRE, an intermediate model for embedded systems with asynchrony and quantitative time [33], [30], [31]. We also undertook the development of an automated translator from FIACRE to LOTOS (see § 5.2.3).
- We wrote a survey [37] of several action-based temporal logics (HML, ACTL, PDL, modal μ-calculus) and their various extensions, which are mostly used for specifying the properties of concurrent asynchronous systems described in languages with labeled transition system semantics, such as FIACRE.
- In collaboration with colleagues from LAAS-CNRS, we proposed techniques to fight against state explosion [36] which could arise when verifying FIACRE models.
- We participated in the TOPCASED Quality Group, which defines the quality policy for TOPCASED (in particular, a set of mandatory requirements) and evaluates the TOPCASED development activities.

H. Garavel is the INRIA representative at the TOPCASED executive committee, for which he served as the secretary during the elaboration phase of the TOPCASED proposal.

7. Other Grants and Activities

7.1. National Collaborations

The VASY project-team plays an active role in the joint research center launched in 2004 between INRIA Rhône-Alpes and the LETI laboratory of CEA-Grenoble. In collaboration with LETI scientists (Edith Beigné, François Bertrand, Fabien Clermidy, Virang Shah, Yvain Thonnart, and Pascal Vivet), VASY develops software tools for the design of asynchronous circuits and architectures such as GALS (*Globally Asynchronous Locally Synchronous*), NOCs (*Networks on Chip*), and SOCs (*Systems on Chip*). In 2007, this collaboration became part of the MULTIVAL project (see § 6.3), and focused on the CHP2LOTOS translator (see § 5.2.3) and its application to the FAUST architecture (see § 5.3.2).

Together with the OASIS project-team of INRIA Sophia-Antipolis (Antonio Cansado and Eric Madelaine), the LTCI team of ENST-Paris (Irfan Hamid, Elie Najm, and Sylvie Vignes), the SVF team of the LAAS-CNRS laboratory (Bernard Berthomieu, Florent Peres, and François Vernadat), and the MVR team of IRIT (Mamoun Filali), VASY participated to the national action FIACRE (ACI *Sécurité Informatique*) started in 2004 and ended in September 2007 (see http://www-sop.inria.fr/oasis/fiacre). In 2007, we completed the definition of the FIACRE intermediate model (see § 5.2.3).

Additionally, we collaborated in 2007 with several INRIA project-teams:

- ATLAS (Nantes): collaboration in the framework of the OPENEMBEDD national action (Jean Bézivin and Frédéric Jouault);
- ATOLL/ALPAGE (Rocquencourt): enhancements to the SYNTAX V6 compiler generation software (Pierre Boullier, Philippe Deschamp, and Benoît Sagot);
- ESPRESSO (Rennes): collaboration in the framework of the TOPCASED and OPENEMBEDD national actions (Jean-Pierre Talpin and Juan Peralta);
- HELIX (Rhône-Alpes): applications of model checking to biological systems (Estelle Dumas, Hidde de Jong, Pedro Monteiro, Michel Page, and Adrien Richard);
- OASIS (Sophia-Antipolis): collaboration in the framework of the FIACRE national action (Antonio Cansado and Eric Madelaine).

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

- LAAS-CNRS laboratory (Toulouse): collaboration in the framework of the FIACRE, OPENEMBEDD, and TOPCASED projects (Bernard Berthomieu and François Vernadat);
- LE2I laboratory (Dijon): since December 2006, R. Mateescu, E. Oudot, A. Wijs, and M. Zidouni are hosted by Université de Bourgogne.

7.2. International Collaborations

The VASY project-team of INRIA and the SEN2 team of CWI collaborate in SENVA, a joint research team on safety-critical systems (see http://www.inrialpes.fr/vasy/senva). Launched in 2004, the SENVA team is supported by INRIA's European and International Affairs Department and by CWI. The first three years of SENVA were favorably evaluated by a panel of international experts in November 2006. In 2007, the SENVA collaboration terminated as most SEN2 members moved from CWI to the University of Twente.

The VASY project-team is member of the FMICS (*Formal Methods for Industrial Critical Systems*) working group of ERCIM (see http://www.inrialpes.fr/vasy/fmics). From July 1999 to July 2001, H. Garavel chaired this working group. Since July 2002, he is member of the FMICS Board, in charge of dissemination actions. Within FMICS, R. Mateescu contributes to the preparation of a "Formal Methods Handbook".

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, launched in 2005 and chaired by Luca Aceto.

In addition to our partners in aforementioned contractual collaborations, we had scientific relations in 2007 with several international universities and research centers, including:

- Imperial College (Jeff Kramer and Jeff Magee),
- LIAMA, China (Claude Helmstetter and Vania Joloboff),
- Polytechnic University of Bucharest (Valentin Cristea),
- Saarland University (Pepijn Crouzen, Holger Hermanns, Sven Johr, and Reza Pulungan),
- University of Málaga (Gwen Salaün and David Sanán),
- University of Malta (Gordon Pace and Sandro Spina), and
- University of Twente (Stefan Blom, Jaco van de Pol, and Michael Weber).

7.3. Visits and Invitations

In 2007, we had the following scientific exchanges:

- David Sanán (PhD student, University of Málaga, Spain) visited us from March 1st to May 31, 2007. During his visit, he focused on enhancing the C.OPEN tool for translating multithreaded C programs with dynamic data types into the implicit graphs defined by OPEN/CÆSAR.
- Pepijn Crouzen (Saarland University, Saarbrücken, Germany) visited us on June 19–20, 2007.
- The second EC-MOAN project meeting was held at INRIA Grenoble – Rhône-Alpes on September 3–4, 2007. In addition to R. Mateescu and E. Oudot from the VASY project-team, Hidde de Jong, Delphine Ropers, Valentina Baldazzi, Pedro Monteiro, and Estelle Dumas (HELIX projet team), Jaco van de Pol and Stefan Blom (University of Twente), Frank Bruggeman and Fred Boogerd (Free University of Amsterdam), Anton Wijs and Jan van Schuppen (CWI Amsterdam), Hongwu Ma (University of Edinburgh), Lubos Brim, Jiri Barnat, Ivana Cerna, and David Safranek (Masaryk University Brno) attended this meeting.
- Sandro Spina (University of Malta, Malta) visited us from September 3 to October 24, 2007.

8. Dissemination

8.1. Software Dissemination and Internet Visibility

The VASY project-team distributes two main software tools: the CADP toolbox (see § 4.1) and the TRAIAN compiler (see § 4.2). In 2007, the main facts are the following:

- We prepared an updated version named CADP2006-a "Edinburgh", released on July 26, 2007, which corrects 4 problems encountered on LINUX and WINDOWS architectures. We also prepared and distributed 2 successive beta-versions (2007-a and 2007-b) of CADP.
- The number of license contracts signed for CADP increased from 366 to 394.
- We were requested to grant CADP licenses for 589 different computers in the world.
- The TRAIAN compiler was downloaded by **46** different sites.

The VASY WEB site (see http://www.inrialpes.fr/vasy) was regularly updated with scientific contents, announcements, publications, etc.

In September 2007, we opened the "CADP Forum" (see http://www.inrialpes.fr/vasy/cadp/forum.html) for discussions regarding the CADP toolbox. By the end of December 2007, this forum had 41 registered users and 67 messages (questions and answers).

In 2007, we revised our internal workflow for distributing CADP, so as to automate the licensing process (which was needed to support a growing user base) and to improve its overall reactivity and quality.

8.2. Program Committees

In 2007, the members of VASY assumed the following responsibilities:

- F. Lang and W. Serwe were selection committee members of a special issue of JESA (*Journal Européen des Systèmes Automatisés*) on *Formal Approaches for Specification and Verification of Real-time Systems*, to appear in 2008.
- H. Garavel is a steering committee member of the PDMC (*Parallel and Distributed Methods in Verification*) series of international workshops.
- H. Garavel was a program committee member of TACAS'2007 (*13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*), Porto, Portugal, March 24–April 1st, 2007.
- F. Lang was a program committee member of NEPTUNE'2007 (*Nice Environment with a Process and Tools Using Norms and Examples*), CNAM Paris, France, May 22–23, 2007.
- H. Garavel was a program committee member of FORTE'2007 (*27th IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems*), Tallinn, Estonia, June 26–29, 2007.
- R. Mateescu was a program committee member of FMICS'2007 (*12th International Workshop on Formal Methods for Industrial Critical Systems*), Berlin, Germany, July 1-2, 2007.
- R. Mateescu was a program committee member of PDMC'2007 (*6th International Workshop on Parallel and Distributed Methods in Verification*), Berlin, Germany, July 8, 2007.
- R. Mateescu was a program committee member of ICSEA'2007 (*2nd International Conference on Software Engineering Advances*, Cap Esterel, France, August 25-31, 2007).
- F. Lang was a scientific committee member of ETR'2007 (*5ème Ecole d'été Temps-Réel*), Nantes, France, September 3–7, 2007.
- F. Lang was a program committee member of ECSA'2007 (*1st European Conference on Software Architectures*), Madrid, Spain, September 24–26, 2007.
- R. Mateescu was an organization committee member of MSR'2007 (*6ème Colloque Francophone sur la Modélisation des Systèmes Réactifs*), Lyon, France, October 17–19, 2007.

8.3. Lectures and Invited Conferences

In 2007, we gave talks in several international conferences and workshops (see bibliography below). Additionally:

- H. Garavel and R. Lacroix visited the ATOLL/ALPAGE project-team at INRIA Rocquencourt on January 15, 2007.
- R. Mateescu gave a talk entitled "MCL: *A Model Checking Language for Concurrent Value-Passing Systems*" at the LIP laboratory (Lyon, France) on January 18, 2007.
- F. Lang participated to the TOPCASED workshop held at LAAS-CNRS (Toulouse, France) on January 23–24, 2007. He gave, jointly with François Vernadat (LAAS-CNRS), a talk entitled "*Le langage pivot asynchrone* FIACRE *et les méthodes formelles*".
- O. Ponsini participated to the "*Workshop on Certification SoC & IP*" at MINALOGIC (Grenoble, France) on January 25, 2007.
- VASY organized the 1st quarterly MULTIVAL meeting, held at INRIA Rhône-Alpes (France) on March 1st, 2007. H. Garavel gave a talk entitled "*Almost Ten Years of Process Algebras and Model Checking for Multiprocessor Architectures*".
- F. Lang gave a talk entitled "*Le langage pivot asynchrone* FIACRE" at INRIA Sophia-Antipolis (France) on March 7–8, 2007.
- O. Ponsini participated to a training on "TLM *modeling in* SYSTEMC" given by STMICROELECTRONICS (Grenoble, France) on March 12–16, 2007.
- VASY organized the 2nd semestrial OPENEMBEDD meeting, held at INRIA Rhône-Alpes (France) on April 4–5, 2007. N. Lépy gave a talk entitled "*An interface for* CADP *using* ECLIPSE" on April 4, 2007. F. Lang gave a talk entitled "*Le langage pivot asynchrone* FIACRE" on April 5, 2007.
- O. Ponsini gave a talk entitled "*Deux approches pour l'utilisation pratique des méthodes de vérification formelle*" at the LIP6 laboratory (Paris, France) on May 4, 2007.
- H. Garavel and Laurent Julliard gave a presentation of the "*pôle de compétitivité Minalogic*" at INRIA Rhône-Alpes in the series "*Une heure ensemble*" on May 14, 2007.
- F. Lang participated to the NEPTUNE conference held at CNAM (Paris, France) on May 22–23, 2007. He participated to a debate entitled "*Comment optimiser les développements par rapport aux besoins dans le cadre des pôles de compétitivité ?*".
- H. Garavel organized the visit of the SAVE-IT industrial graduate school (Mälardalen, Sweden) at INRIA Rhône-Alpes on June 6, 2007. He gave a talk entitled "CADP*: Software Tools for System-Level Verification of Asynchronous Designs*". W. Serwe gave a demonstration of the CADP toolbox for the verification of asynchronous hardware.
- D. Champelovier, N. Coste, H. Garavel, H. Hermanns, R. Lacroix, O. Ponsini, W. Serwe, M. Vidal, and M. Zidouni participated to the 2nd quarterly MULTIVAL meeting held at CEA/LETI (Grenoble, France) on June 14, 2007. H. Garavel gave a talk entitled "*Migrating* CADP *to 64-bit machines*". O. Ponsini gave a talk entitled "*Translating* SYSTEMC/TLM *models into* LOTOS".
- H. Garavel participated to the OPEES workshop in Paris on June 28–29, 2007. He gave a talk entitled "*Le projet Multival*".
- J. Stoecker participated to ETR'07 (*5ème Ecole d'été Temps-Réel*), Nantes, France, September 3–7, 2007.
- H. Garavel participated to the IFIP workshop on "*Applying Concurrency Research in Industry*" held in Lisbon (Portugal) on September 7, 2007. He gave a talk entitled "CADP *Twenty Years After*".
- D. Champelovier, N. Coste, H. Garavel, H. Hermanns, R. Mateescu, O. Ponsini, W. Serwe, and M. Zidouni participated to the 3rd quarterly MULTIVAL meeting held at BULL (Paris, France) on September 20, 2007.

- F. Lang gave a talk entitled "*Recent developments and improvements of the* CADP *toolbox*" at France Telecom R&D (Lannion) on September 24, 2007.
- D. Thivolle participated to the technical meeting of engineers involved in the OPENEMBEDD project held in Rennes on October 3–4, 2007.
- H. Garavel participated to a presentation of the collaborative project MULTIVAL held at STMICROELECTRONICS (Grenoble, France) on November 9, 2007.
- N. Coste and H. Hermanns participated to the workshop "*Two Decades of Probabilistic Verification – Reflections and Perspectives*" held at the Lorentz center (Leiden, The Netherlands) on November 12–16, 2007.
- X. Clerc, H. Garavel, F. Lang, and D. Thivolle participated to the 3rd semestrial OPENEMBEDD meeting held at LAAS-CNRS (Toulouse, France) on November 12–13, 2007.
- X. Clerc, H. Garavel, F. Lang, and D. Thivolle participated to the TOPCASED WP3 meeting held at LAAS-CNRS (Toulouse, France) on November 13–14, 2007. F. Lang gave a talk entitled "*Le langage pivot asynchrone* FIACRE".
- R. Mateescu gave a seminar entitled "*On-the-Fly Model Checking with Evaluator 3.5*" to the MSc computer science students of Université de Bourgogne on November 23, 2007.
- E. Oudot gave a seminar entitled "*Vérification incrémentale de systèmes temporisés à composants*" to the MSc computer science students of Université de Bourgogne on November 23, 2007.
- A. Wijs gave a seminar entitled "*Directed Searching through State Spaces*" to the MSc computer science students of Université de Bourgogne on November 23, 2007.
- N. Coste, H. Garavel, H. Hermanns, W. Serwe, and M. Zidouni participated to the MULTIVAL doctoral workshop held at Saarland University (Saarbrücken, Germany) on November 26–29, 2007. W. Serwe gave a talk entitled "*Modeling and Analysis of Asynchronous Concurrent Hardware*".
- D. Champelovier, N. Coste, H. Garavel, Y. Guerte, R. Hérilier, H. Hermanns, R. Lacroix, R. Mateescu, O. Ponsini, W. Serwe, and M. Zidouni participated to the 4th quarterly MULTIVAL meeting held at STMICROELECTRONICS (Grenoble, France) on December 4, 2007. D. Champelovier gave a talk entitled "*Le langage* LOTOS NT *et le traducteur* LNT2LOTOS".

8.4. Teaching Activities

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

In 2007:

- H. Garavel, F. Lang, and W. Serwe created, jointly with Pascal Raymond (VERIMAG), a course on "*Méthodes Formelles de Développement*" to the computer science engineering students of CNAM (*Conservatoire National des Arts et Métiers*) Grenoble (21 hours).
- F. Lang and W. Serwe gave the course on "*Temps Réel*" to the 3rd year students of ENSIMAG (18 hours).
- E. Oudot and R. Mateescu, together with Sylvain Rampacek (Université de Bourgogne) gave the course on "*Méthodes formelles*" to the 5th year students of ESIREM (86 hours).
- O. Ponsini was a teaching assistant for a tutorial on programming and mathematics given to high-school pupils with the INRIA's MOBINET pedagogic platform during the engineer weeks of the Institut National Polytechnique de Grenoble (3 hours).
- R. Mateescu was a jury member and reviewed Juan José Sánchez Penas' PhD thesis entitled "*From Software Architecture to Formal Verification of a Distributed System*" defended at University of Corunha (Spain) on December 24, 2006.

- H. Garavel supervised the MSc thesis of D. Thivolle entitled "*Etude de la traduction de* CSP$_m$ *en* LOTOS", defended at Université Joseph Fourier (Grenoble) on June 19, 2007.
- H. Garavel and F. Lang supervised the internship (*mémoire* CNAM) of N. Lépy entitled "*Intégration de* CADP *dans l'environnement* ECLIPSE".
- H. Garavel was a jury member of Tarek Sadani's PhD thesis entitled "*Validation de spécifications formelles* RT-LOTOS", defended at ENSICA (INPT, Toulouse) on May 3, 2007.
- R. Mateescu was the jury president of Renaud Vanlande's PhD thesis entitled "*C-DMF : une architecture de modélisation sémantique des données. Application à la gestion technique de patrimoine immobilier*", defended at Université de Bourgogne on December 3, 2007.
- W. Serwe was a jury member of Xing Lu's MSc thesis entitled "*Aspects de sûreté et programmes non déterministes*", defended at université Joseph Fourier (Grenoble) on September 3, 2007.
- F. Lang was a jury member and reviewed Emmanuel Donin de Rosière's PhD thesis entitled "*Un langage non-déterministe pour l'écriture de scénarios de test*", defended at ENSSAT (Lannion) on September 24, 2007.
- R. Mateescu was a jury member and reviewed Anton Wijs's PhD thesis entitled "*What to Do Next? Analysing and Optimising System Behaviour in Time*", defended at the Free University of Amsterdam (Netherlands) on October 2, 2007.
- H. Garavel and W. Serwe were jury members of Jérôme Fereyre's engineer thesis entitled "*Conception et amélioration d'outils logiciels pour la vérification distribuée*", defended at CNAM (Grenoble) on December 20, 2007.
- R. Mateescu was a suppliant member of the "Commission de spécialistes" at Université de Bourgogne (section 27).

8.5. Miscellaneous Activities

D. Champelovier is a member of the computing facilities committee of INRIA Rhône-Alpes.

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

F. Lang participates to the consultative organizational committee of INRIA Rhône-Alpes.

F. Lang led the working group (5 persons) in charge of proposing a new distribution of offices among the research and administrative teams located in the INRIA building of Montbonnot. The task of this working group ended in September 2007, after a moving of several teams.

F. Lang participates to the working group in charge of specifying requirements about the new aisle of the INRIA building of Montbonnot, which should be delivered by the end of 2009.

O. Ponsini contributed to the organization of the LIG laboratory council elections.

9. 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. *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.

[3] 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, Full version available as Inria Research Report RR-3352, vol. 1384, Springer Verlag, March 1998, p. 68–84, http://hal.inria.fr/inria-00073337.

[4] 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, Full version available as Inria Research Report 4492, vol. 2391, Springer Verlag, July 2002, p. 410–429, http://hal.inria.fr/inria-00072096.

[5] 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), Full version available as Inria Research Report RR-4223, Kluwer Academic Publishers, IFIP, August 2001, p. 377–392, http://hal.inria.fr/inria-00072396.

[6] 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, Full version available as Inria Research Report RR-4666, vol. 2529, Springer Verlag, November 2002, p. 276–291, http://hal.inria.fr/inria-00071919.

[7] 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, vol. 2304, Springer Verlag, April 2002, p. 9–13.

[8] 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, Full version available as Inria Research Report RR-4341, vol. 2057, Springer Verlag, May 2001, p. 217–234, http://hal.inria.fr/inria-00072247.

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

[10] 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, IFIP, June 1990, p. 379–394.

[11] 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), Invited talk, CWI, May 1998, p. 187–230.

[12] 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, IFIP, October 1999, p. 185–202.

[13] H. GARAVEL, C. VIHO, M. ZENDRI. *System Design of a CC-NUMA Multiprocessor Architecture using Formal Specification, Model-Checking, Co-Simulation, and Test Generation*, in "Springer International Journal on Software Tools for Technology Transfer (STTT)", Full version available as Inria Research Report RR-4041, vol. 3, n° 3, July 2001, p. 314–331, http://hal.inria.fr/inria-00072597.

[14] R. MATEESCU. *CAESAR_SOLVE: A Generic Library for On-the-Fly Resolution of Alternation-Free Boolean Equation Systems*, in "Springer International Journal on Software Tools for Technology Transfer (STTT)", Full version available as Inria Research Report RR-5948, July 2006, vol. 8, n° 1, February 2006, p. 37–56, https://hal.inria.fr/inria-00084628.

[15] R. MATEESCU, M. SIGHIREANU. *Efficient On-the-Fly Model-Checking for Regular Alternation-Free Mu-Calculus*, in "Science of Computer Programming", vol. 46, n° 3, March 2003, p. 255–281.

[16] G. SALAÜN, W. SERWE. *Translating Hardware Process Algebras into Standard Process Algebras — Illustration with CHP and LOTOS*, in "Proceedings of the 5th International Conference on Integrated Formal Methods IFM'2005 (Eindhoven, The Netherlands)", J. VAN DE POL, J. ROMIJN, G. SMITH (editors), Lecture Notes in Computer Science, Full version available as Inria Research Report RR-5666, vol. 3771, Springer Verlag, November 2005, p. 287–306.

Year Publications

Doctoral dissertations and Habilitation theses

[17] R. MATEESCU. *Composants génériques pour l'analyse des systèmes de transitions*, Habilitation à Diriger les Recherches, September 2007.

Articles in refereed journals and book chapters

[18] F. LANG. *Explaining the Lazy Krivine Machine Using Explicit Substitution and Addresses*, in "Journal of Higher-Order and Symbolic Computation, special issue on Krivine's machine", vol. 20, n° 3, September 2007, p. 257–270.

Publications in Conferences and Workshops

[19] B. BERTHOMIEU, J.-P. BODEVEIX, P. FARAIL, M. FILALI, H. GARAVEL, P. GAUFILLET, F. LANG, F. PERES, R. SAAD, J. STOECKER, F. VERNADAT. *FIACRE: an Intermediate Language for Model Verification in the TOPCASED Environment*, in "Proceedings of the 4th European Congress on Embedded Real-Time Software ERTS'08 (Toulouse, France)", J.-C. LAPRIE (editor), to appear, SIA (the French Society of Automobile Engineers), AAAF (the French Society of Aeronautic and Aerospace), and SEE (the French Society for Electricity, Electronics, and Information & Communication Technologies), January 2008.

[20] N. COSTE, H. GARAVEL, H. HERMANNS, R. HERSEMEULE, Y. THONNART, M. ZIDOUNI. *Quantitative Evaluation in Embedded System Design: Validation of Multiprocessor Multithreaded Architectures*, in "Special Session at Design, Automation & Test in Europe DATE'08 (Munich, Germany)", to appear, March 2008.

[21] 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, to appear, 2008.

[22] 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, vol. 4590, Springer Verlag, July 2007, p. 158–163.

[23] R. MATEESCU, P. POIZAT, G. SALAÜN. *Behavioral Adaptation of Component Compositions based on Process Algebra Encodings*, in "Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering ASE'2007 (Atlanta, Georgia, USA)", A. EGYED, B. FISCHER (editors), Full version available as INRIA Research Report RR-6362, ACM Press, November 2007, p. 385–388.

[24] 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)", to appear, May 2008.

[25] 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)", to appear, May 2008.

[26] G. SALAÜN, J. KRAMER, F. LANG, J. MAGEE. *Translating FSP into LOTOS and Networks of Automata*, in "Proceedings of the 6th International Conference on Integrated Formal Methods IFM'2007 (Oxford, United Kingdom)", J. DAVIES, W. SCHULTE, J. S. DONG (editors), Lecture Notes in Computer Science, Springer Verlag, July 2007.

[27] G. SALAÜN, W. SERWE, Y. THONNART, P. VIVET. *Formal Verification of CHP Specifications with CADP — Illustration on an Asynchronous Network-on-Chip*, in "Proceedings of the 13th IEEE International Symposium on Asynchronous Circuits and Systems ASYNC 2007 (Berkeley, California, USA)", IEEE Computer Society Press, March 2007, p. 73–82.

Internal Reports

[28] B. BERTHOMIEU, J.-P. BODEVEIX, M. FILALI, H. GARAVEL, F. LANG, F. PERES, R. SAAD, J. STOECKER, F. VERNADAT. *Spécification des besoins pour les activités de vérification*, Project deliverable 4.2.2, ANR 05 RNTL 03101 project OpenEmbedd, September 2007.

[29] B. BERTHOMIEU, J.-P. BODEVEIX, M. FILALI, H. GARAVEL, F. LANG, F. PERES, R. SAAD, J. STOECKER, F. VERNADAT. *The Syntax and Semantics of Fiacre – version 1.0 alpha*, Project deliverable 4.2.4, ANR 05 RNTL 03101 project OpenEmbedd, May 2007.

[30] B. BERTHOMIEU, J.-P. BODEVEIX, M. FILALI, H. GARAVEL, F. LANG, F. PERES, R. SAAD, J. STOECKER, F. VERNADAT. *The Syntax and Semantics of Fiacre – version 1.0 alpha*, Project deliverable F3.2.2, AESE (pôle de compétitivité mondial Midi-Pyrénées & Aquitaine: Aéronautique, Espace et Systèmes Embarqués) project Topcased, July 2007.

[31] B. BERTHOMIEU, J.-P. BODEVEIX, M. FILALI, H. GARAVEL, F. LANG, F. PERES, R. SAAD, J. STOECKER, F. VERNADAT. *The Syntax and Semantics of Fiacre – version 1.0 beta*, Project deliverable F3.2.2,

AESE (pôle de compétitivité mondial Midi-Pyrénées & Aquitaine: Aéronautique, Espace et Systèmes Embarqués) project Topcased, September 2007.

[32] B. BERTHOMIEU, M. FILALI, H. GARAVEL, F. LANG, F. PERES, J. STOECKER, F. VERNADAT. *Spécification des besoins pour la représentation du temps quantitatif dans un modèle de calcul asynchrone*, Project deliverable 4.2.1, ANR 05 RNTL 03101 project OpenEmbedd, September 2007.

[33] B. BERTHOMIEU, M. FILALI, F. LANG, F. PERES, F. VERNADAT, J. STOECKER. *Sélection des outils de vérification/simulation à connecter*, Project deliverable F3.2.1, AESE (pôle de compétitivité mondial Midi-Pyrénées & Aquitaine: Aéronautique, Espace et Systèmes Embarqués) project Topcased, February 2007.

[34] M. FILALI, F. LANG, F. PERES, J. STOECKER, F. VERNADAT. *Modèles pivots pour la représentation des processus concurrents asynchrones*, Project deliverable 4.2.3, ANR 05 RNTL 03101 project OpenEmbedd, February 2007.

[35] M. FILALI, F. LANG, F. PERES, F. VERNADAT. *Recommandations pour les spécifications des langages pivots*, Project deliverable F3.1, AESE (pôle de compétitivité mondial Midi-Pyrénées & Aquitaine: Aéronautique, Espace et Systèmes Embarqués) project Topcased, February 2007.

[36] H. GARAVEL, F. LANG, F. VERNADAT. *Etat de l'art: Techniques de lutte contre l'explosion combinatoire – version 1.0 alpha*, Project deliverable F3.7.1, AESE (pôle de compétitivité mondial Midi-Pyrénées & Aquitaine: Aéronautique, Espace et Systèmes Embarqués) project Topcased, March 2007.

[37] R. MATEESCU. *Temporal Logics for the Specification of Concurrent Systems – version 1.0 alpha*, Project deliverable F3.10.1, AESE (pôle de compétitivité mondial Midi-Pyrénées & Aquitaine: Aéronautique, Espace et Systèmes Embarqués) project Topcased, November 2007.

Miscellaneous

[38] J. FEREYRE. *Conception et amélioration d'outils logiciels pour la vérification distribuée*, Mémoire d'ingénieur, CNAM, Grenoble, December 2007.

[39] H. GARAVEL. *CADP 2006: Modelling and Checking Asynchronous Systems*, INédit 58, INRIA, March 2007, http://www.inria.fr/actualites/inedit/inedit58_sommaire.en.html.

[40] H. GARAVEL. *The Future is Already Asynchronous*, INédit 60, INRIA, July 2007, http://www.inria.fr/actualites/inedit/inedit60_sommaire.en.html.

[41] F. LANG, X. CLERC. *Translating FIACRE into LOTOS*, Inria/Vasy, 32 pages, 2007.

[42] F. LANG, G. SALAÜN, R. HÉRILIER, J. KRAMER, J. MAGEE. *Translating FSP into LOTOS and Networks of Automata*, Revised version. Inria/Vasy, 27 pages, 2007.

[43] THE VASY TEAM. *The CADP Newsletter – Nr. 6*, April 2007, http://www.inrialpes.fr/vasy/cadp/news6.html.

[44] D. THIVOLLE. *Etude de la traduction de CSP_m en LOTOS*, Mémoire de master 2 recherche, Université Joseph Fourier, Grenoble, September 2007.

References in notes

[45] B. K. AICHERNIG, M. WEIGLHOFER, B. PEISCHL, F. WOTAWA. *Test Purpose Generation in an Industrial Application*, in "3rd International Workshop on Advances in Model-Based Testing", ACM, July 2007, p. 115–125.

[46] E. BEIGNÉ, F. CLERMIDY, P. VIVET, A. CLOUARD, M. RENAUDIN. *An Asynchronous NoC Architecture Providing Low Latency Service and its Multi-Level Design Framework*, in "Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems ASYNC'05 (New York, USA)", IEEE Computer Society Press, March 2005, p. 54–63.

[47] B. BERTHOMIEU, P. RIBET, F. VERNADAT, J. BERNARTT, J.-M. FARINES, J.-P. BODEVEIX, M. FILALI, G. PADIOU, P. MICHEL, P. FARAIL, P. GAUFILLET, P. DISSAUX, J.-L. LAMBERT. *Towards the verification of real-time systems in avionics: the COTRE approach*, in "Proceedings of the 8th International Workshop on Formal Methods for Industrial Critical Systems FMICS'2003 (Trondheim, Norway)", T. ARTS, W. FOKKINK (editors), Electronic Notes on Theoretical Computer Science, vol. 80, Elsevier, June 2003, p. 201–216.

[48] S. BLOM, J. R. CALAMÉ, B. LISSER, S. ORZAN, J. PANG, J. VAN DE POL, M. T. DASHTI, A. J. WIJS. *Distributed Analysis with μCRL: A Compendium of Case Studies*, in "13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS'2007 (Braga, Portugal)", O. GRUMBERG, M. HUTH (editors), Lecture Notes in Computer Science, vol. 4424, Springer Verlag, March 2007, p. 683–689.

[49] G. CHEHAIBAR, H. GARAVEL, L. MOUNIER, N. TAWBI, F. ZULIAN. *Specification and Verification of the PowerScale Bus Arbitration Protocol: An Industrial Experiment with LOTOS*, 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'96 (Kaiserslautern, Germany)", R. GOTZHEIN, J. BREDEREKE (editors), Full version available as Inria Research Report RR-2958, Chapman & Hall, IFIP, October 1996, p. 435–450, http://hal.inria.fr/inria-00073740.

[50] 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", vol. 8, n^{o} 2, April 1986, p. 244–263.

[51] J. CUBO, G. SALAÜN, C. CANAL, E. PIMENTEL, P. POIZAT. *A Model-Based Approach to the Verification and Adaptation of WF/.NET Components*, in "Proceedings of the 4th International Workshop on Formal Aspects of Component Software", Electronic Notes in Theoretical Computer Science, Elsevier, September 2007.

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

[53] A. DESMOULIN, C. VIHO. *A New Method for Interoperability Test Generation*, in "19th IFIP International Conference on Testing of Software and Communicating Systems and 7th International Workshop on Formal Approaches to Testing of Software", A. PETRENKO, M. VEANES, J. TRETMANS, W. GRIESKAMP (editors), Lecture Notes in Computer Science, vol. 4581, Springer Verlag, June 2007, p. 58–73.

[54] MARÍA-DEL-MAR. GALLARDO, C. JOUBERT, P. MERINO. *On-the-Fly Data Flow Analysis based on Verification Technology*, in "Proceedings of the 6th International Workshop on Compiler Optimization meets

Compiler Verification", R. DRECHSLER, S. GLESNER, J. KNOOP (editors), Electronic Notes in Theoretical Computer Science, vol. 190, Elsevier, March 2007, p. 33–48.

[55] MARÍA-DEL-MAR. GALLARDO, C. JOUBERT, P. MERINO, D. SANÁN. *C.OPEN and ANNOTATOR: Tools for On-the-Fly Model Checking C Programs*, in "Proceedings of the 14th International SPIN Workshop on Model Checking of Software", D. BOSNACKI, S. EDELKAMP (editors), Lecture Notes in Computer Science, vol. 4595, Springer Verlag, July 2007, p. 268–273.

[56] MARÍA-DEL-MAR. GALLARDO, C. JOUBERT, P. MERINO, D. SANÁN. *On-the-Fly API Influence Analysis of Software*, in "Proceedings of the 2nd International Conference on Science and Technology", P. MERINO, M. BAKKALI (editors), Spicum, March 2007.

[57] MARÍA-DEL-MAR. GALLARDO, C. JOUBERT, P. MERINO, D. SANÁN. *On-the-Fly Model Checking for C Programs with Extended CADP in FMICS-jETI*, in "Proceedings of the 12th IEEE International Conference on Engineering of Complex Computer Systems", IEEE Computer Society Press, July 2007, p. 321–329.

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

[59] Q. GUO, J. DERRICK. *Verification of Timed Erlang/OTP Components using the Process Algebra μCRL*, in "2007 SIGPLAN Workshop on Erlang", ACM, October 2007, p. 55–64.

[60] Q. GUO. *Verifying Erlang/OTP Components in μCRL*, in "Proceedings of the 27th IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems FORTE 2007 (Tallinn, Estonia)", J. DERRICK, J. VAIN (editors), Lecture Notes in Computer Science, vol. 4574, Springer Verlag, June 2007, p. 227–246.

[61] I. HAMID, E. NAJM. *Real-time Connectors for Deterministic Data-Flow*, in "13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications", IEEE Computer Society, August 2007, p. 173-182.

[62] C. HELMSTETTER. *Validation de modèles de systèmes sur puce en présence d'ordonnancements indéterministes et de temps imprécis*, Thèse de Doctorat, Institut National Polytechnique de Grenoble, March 2007.

[63] C. HELMSTETTER, F. MARANINCHI, L. MAILLET-CONTOZ, M. MOY. *Automatic Generation of Schedulings for Improving the Test Coverage of Systems-on-a-Chip*, in "Proceedings 6th International Conference on Formal Methods in Computer-Aided Design FMCAD 2006 (San Jose, California, USA)", November 2006, p. 171–178.

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

[65] H. HOJJAT, M. SIRJANI, M. R. MOUSAVI, J. F. GROOTE. *Sarir: A Rebeca to mCRL2 Translator*, in "Proceedings of the Seventh International Conference on Application of Concurrency to System Design", IEEE Computer Society, July 2007, p. 216–222.

[66] IEEE. *IEEE Standard SystemC Language Reference Manual*, IEEE Standard, n^{o} 1666-2005, Institution of Electrical and Electronic Engineers, December 2005.

[67] 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, Extended version with proofs available as Research Report VERIMAG RR97-01, vol. 1217, Springer Verlag, April 1997.

[68] C. MAASBOMMEL. *A Formal Analysis of the RIES Internet Voting Protocol*, Technical report, Free University of Amsterdam, The Netherlands, February 2007.

[69] J. MAGEE, J. KRAMER. *Concurrency: State Models and Java Programs*, Wiley, 1999.

[70] J. PANG, W. FOKKINK, R. HOFMAN, R. VELDEMA. *Model Checking a Cache Coherence Protocol of a Java DSM Implementation*, in "Journal of Logic and Algebraic Programming", vol. 42, n^{o} 1, March 2007, p. 1–43.

[71] B. PLOEGER, L. SOMERS. *Analysis and Verification of an Automatic Document Feeder*, in "Proceedings of the 21st ACM Symposium on Applied Computing SAC 2007 (Seoul, Korea)", ACM, March 2007, p. 1499–1505.

[72] P. POIZAT, G. SALAÜN. *Adaptation of Open Component-Based Systems*, in "9th International Conference on Formal Methods for Open Object-Based Distributed Systems", M. M. BONSANGUE, E. B. JOHNSEN (editors), Lecture Notes in Computer Science, vol. 4468, Springer Verlag, June 2007, p. 141–156.

[73] M. RENAUDIN. *TAST Compiler and TAST-CHP Language – Version 0.6*, TIMA Laboratory, CIS Group, 2005.

[74] N. S. ROSA, P. R. F. CUNHA. *Using LOTOS for Formalising Wireless Sensor Network Applications*, in "Sensors", vol. 7, n^{o} 8, August 2007, p. 1447–1461.

[75] A. W. ROSCOE, C. A. R. HOARE, R. BIRD. *The Theory and Practice of Concurrency*, Prentice Hall, 1997.

[76] A. ROSE, S. SWAN, J. PIERCE, J.-M. FERNANDEZ. *Transaction Level Modeling in SystemC*, Open SystemC Initiative, 2005.

[77] B. SCATTERGOOD. *The Semantics and Implementation of Machine-Readable CSP*, PhD Thesis, Oxford University Computing Laboratory, 1998.

[78] C. TRAULSEN, J. CORNET, M. MOY, F. MARANINCHI. *A SystemC/TLM semantics in Promela and its possible applications*, in "14th International SPIN Workshop on Model Checking Software", Lecture Notes in Computer Science, vol. 4595, Springer Verlag, July 2007, p. 204–222.

[79] M. WEIGLHOFER. *Abstract Test Case Execution*, http://www.ist.tugraz.at/staff/weiglhofer/projects/testexec/index.html.

[80] M. YOELI, R. KOL. , A. Y. ZOMAYA (editor)*Verification of Systems and Circuits Using LOTOS, Petri Nets, and CCS*, Parallel and Distributed Computing, Wiley, March 2008.

[81] J. VAN EIJCK, S. ORZAN. *Epistemic Verification of Anonymity*, in "2nd International Workshop on Views on Designing Complex Architectures", M. TER BEEK, F. GADDUCCI (editors), Electronic Notes in Theoretical Computer Science, vol. 168, Elsevier, September 2007, p. 159–174.