INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team VASY

Validation of Systems

Grenoble - Rhône-Alpes

Theme : Embedded and Real Time Systems

Activity Report

2009

Table of contents

VASY *is an* INRIA *project-team gathering scientists from several European countries.* VASY *is based in Grenoble with members also located at the Polytechnic University of Bucharest (Romania) and Saarland University (Germany). Since January 2007, the* VASY *scientists in Grenoble are also members of the* LIG *joint research laboratory of Centre National de Recherche Scientifique, Grenoble* INP*, and Université Joseph Fourier.*

1. Team

Research Scientist

Hubert Garavel [DR2 INRIA, Team Leader]
Frédéric Lang [CR1 INRIA]
Radu Mateescu [CR1 INRIA]
Wendelin Serwe [CR1 INRIA]

Faculty Member

Gwen Salaün [Associate Professor, Grenoble INP, since September 1, 2009]

External Collaborator

Holger Hermanns [Saarland University]
Etienne Lantreibecq [STMICROELECTRONICS]

Technical Staff

Iker Bellicot [since October 26, 2009]
Simon Bouland [since February 2, 2009]
Yann Genevois [since September 1st, 2009]
Yves Guerte [until August 31, 2009]
Rémi Hérilier
Alain Kaufmann [since October 15, 2009]
Christine McKinty [since February 16, 2009]
Jeanne Merle [until August 31, 2009]
Louis Paternault [until September 30, 2009]
Gideon Smeding [from March 9 to November 30, 2009]

PhD Student

Nicolas Coste [STMICROELECTRONICS, CIFRE grant]
Jan Stoecker [CORDI grant, until August 31, 2009]
Damien Thivolle [MESR grant]
Meriem Zidouni [BULL, CIFRE grant]

Post-Doctoral Fellow

Claude Helmstetter [until November 16, 2009]
Anton Wijs [until November 30, 2009]

Administrative Assistant

Helen Pouchot

Other

Julien Henry [Student intern, from June 16 to July 31, 2009]
Sébastien Meriot [Student intern, from April 6 to July 21, 2009]

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*) [7].

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: Iker Bellicot, Simon Bouland, Hubert Garavel [contact person], Yann Genevois, Yves Guerte, Claude Helmstetter, Rémi Hérilier, Alain Kaufmann, Frédéric Lang, Radu Mateescu, Christine McKinty, Jeanne Merle, Louis Paternault, Gideon Smeding, Wendelin Serwe, Anton Wijs, Damien Thivolle.

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

- CÆSAR.ADT [3] is a compiler that translates LOTOS abstract data types into C types and C functions. The translation involves pattern-matching compiling techniques and automatic recognition of usual types (integers, enumerations, tuples, etc.), which are implemented optimally.
- CÆSAR [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 [4] is a generic software environment for developing tools that explore graphs on the fly (for instance, simulation, verification, and test generation tools). Such tools can be developed independently of any particular high level language. In this respect, OPEN/CÆSAR plays a central role in CADP by connecting language-oriented tools with model-oriented tools. OPEN/CÆSAR consists of a set of 16 code libraries with their programming interfaces, such as:
  - CAESAR_GRAPH, which provides the programming interface for graph exploration,
  - CAESAR_HASH, which contains several hash functions,
  - CAESAR_SOLVE, which resolves boolean equation systems on the fly,
  - CAESAR_STACK, which implements stacks for depth-first search exploration,
  - CAESAR_TABLE, which handles tables of states, transitions, labels, etc.

A number of tools have been developed within the OPEN/CÆSAR environment, among which:

- BISIMULATOR, which checks bisimulation equivalences and preorders,
- CUNCTATOR, which performs on-the-fly steady-state simulation of continuous-time Markov chains,
- DETERMINATOR, which eliminates stochastic nondeterminism in normal, probabilistic, or stochastic systems,
- DISTRIBUTOR, which generates the graph of reachable states using several machines,
- EVALUATOR, which evaluates regular alternation-free μ-calculus formulas,
- EXECUTOR, which performs random execution,
- EXHIBITOR, which searches for execution sequences matching a given regular expression,
- GENERATOR, which constructs the graph of reachable states,
- PROJECTOR, which computes abstractions of communicating systems,
- REDUCTOR, which constructs and minimizes the graph of reachable states modulo various equivalence relations,
- SIMULATOR, XSIMULATOR, and OCIS, which allow interactive simulation, and
- TERMINATOR, which searches for deadlock states.

• BCG (*Binary Coded Graphs*) is both a file format for storing very large graphs on disk (using efficient compression techniques) and a software environment for handling this format. BCG also plays a key role in CADP as many tools rely on this format for their inputs/outputs. The BCG environment consists of various libraries with their programming interfaces, and of several tools, such as:

  - BCG_DRAW, which builds a two-dimensional view of a graph,
  - BCG_EDIT, which allows to modify interactively the graph layout produced by BCG_DRAW,
  - BCG_GRAPH, which generates various forms of practically useful graphs,
  - BCG_INFO, which displays various statistical information about a graph,
  - BCG_IO, which performs conversions between BCG and many other graph formats,
  - BCG_LABELS, which hides and/or renames (using regular expressions) the transition labels of a graph,
  - BCG_MERGE, which gathers graph fragments obtained from distributed graph construction,
  - BCG_MIN, which minimizes a graph modulo strong or branching equivalences (and can also deal with probabilistic and stochastic systems),
  - BCG_STEADY, which performs steady-state numerical analysis of (extended) continuous-time Markov chains,
  - BCG_TRANSIENT, which performs transient numerical analysis of (extended) continuous-time Markov chains, and
  - XTL (*eXecutable Temporal Language*), which is a high level, functional language for programming exploration algorithms on BCG graphs. XTL provides primitives to handle states, transitions, labels, *successor* and *predecessor* functions, etc.

For instance, one can define recursive functions on sets of states, which allow to specify in XTL evaluation and diagnostic generation fixed point algorithms for usual temporal logics (such as HML [59], CTL [50], ACTL [54], 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 LNT2LOTOS tool provides users of LOTOS NT with an alternative to the TRAIAN compiler, by translating a LOTOS NT specification into LOTOS code. The CADP tools can then be applied to this code.

The CADP toolbox also includes 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 [6] scripting language. Both EUCALYPTUS and SVL provide users with an easy and uniform access to the CADP tools by performing file format conversions automatically whenever needed and by supplying appropriate command-line options as the tools are invoked.

4.2. The TRAIAN Compiler

Participants: 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 [8]. The recent compilers developed by the VASY project-team — namely AAL, CHP2LOTOS (see § 5.2.3), EVALUATOR 4.0 (see § 5.1.6), 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 BCG Format and Libraries

Participants: Hubert Garavel, Alain Kaufmann, Frédéric Lang, Radu Mateescu, Louis Paternault, Wendelin Serwe.

BCG (*Binary-Coded Graphs*) is both a file format for the representation of explicit graphs and a collection of libraries and programs dealing with this format.

In 2009, we fixed 13 bugs in the BCG_READ/BCG_WRITE libraries and the BCG_DRAW, BCG_GRAPH, BCG_INFO, BCG_IO, and BCG_LABELS tools, and brought various enhancements such as better diagnostics for BCG_IO and BCG_STEADY, adjustable paper size for BCG_DRAW and BCG_EDIT, faster execution of BCG_INFO, alphabetical sorting of labels in the BCG monitor, etc.

We significantly reduced the size of BCG files when encoding graphs that have many different transition labels. By example, for a graph with one million transitions and one million different labels, the size of BCG file decreased from $156,240$ down to $45,184$ kbytes. The BCG format itself was not modified: only the way of encoding labels changed. Care was taken to preserve total backward compatibility.

We completed the port of the BCG format, libraries, and tools to 64-bit architectures. So doing, the version 1.0 of the BCG format (almost unchanged since 1993) was replaced by a new version 1.1, which can handle graphs up to 2^{44} states and transitions, compared to 2^{29} for the former version 1.0. This format change is backward compatible and transparent to end-users, excepted a small size increase for the BCG files (0.33% on the average, 4.17% in the worst case observed).

Besides, we continued our research on the development of a new version 2.0 of the BCG format that would further increase the compactness of the format. We experimented various compression schemes that could be used in BCG 2.0 on a collection of 250 very large BCG graphs representing the behaviour of real-life systems, which will eventually enrich the existing VLTS (*Very Large Transition Systems*) benchmark suite[1].

5.1.2. The OPEN/CÆSAR Libraries

Participants: Hubert Garavel, Radu Mateescu, Wendelin Serwe, Anton Wijs.

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

In 2009, we improved the CÆSAR_TABLE library in many respects; in particular, the maximal size of each table is now automatically adjusted to the physical memory available (which reduces the initial memory requirements by a factor of up to two) and the internal hash tables are now resized dynamically (reducing, in some large cases, the state space generation from several days down to two hours). We also revised one hash function of the CÆSAR_HASH library to enhance its speed.

We also continued the study of *state space caching* techniques for model checking, which help fighting state explosion by keeping, in a memory cache, only a fixed amount of visited states. When the cache is full and a newly encountered state must be stored, some state present in the cache is removed and replaced with the new one. To support cache management, we developed a prototype library named CÆSAR_CACHE, which allows to build generic, hierarchical caches with a dynamically changing structure. Each cache is organized as a collection of subcaches, each one containing a fixed number of elements and being equipped with its own replacement strategy. The subcaches are linked hierarchically by a parent relation, which defines a directed acyclic graph. This relation is defined when a cache is created and may change dynamically during the usage of the cache.

The CÆSAR_CACHE library currently provides 52 primitives (creation/deletion of caches, insertion and search of elements in a cache, inspection of cache status, display of statistical information about cache usage, etc.). The library offers 15 built-in replacement strategies, among which: LRU/MRU (*Least/Most Recently Used*), LFU/MFU (*Least/Most Frequently Used*), RND (*Random*), etc. The user can also define its own replacement strategy based on specific information that can be attached to each element present in a cache.

[1] http://www.inrialpes.fr/vasy/cadp/resources/benchmark_bcg.html

We also pursued our study, undertaken in 2008, of state space generation using hierarchical adaptive caches. We enhanced our prototype state space generator with the possibility to specify the structure of caches using configuration files, which allow a flexible definition of hierarchical caches with various structures. The experiments we performed on graphs taken from the VLTS benchmark suite indicate that caches organized as trees with two branches, where each branch consists of a stream of subcaches, combine the advantages of each branch considered separately. This work led to a publication in an international conference [29].

5.1.3. The XTL Tool

Participants: Hubert Garavel, Radu Mateescu.

XTL (*eXecutable Temporal Language*, see § 4.1) is both a meta-language and tool allowing to specify temporal logic properties involving data values and to verify them on graphs represented in the BCG format.

In 2009, we enhanced the XTL language to enable the handling of 64-bit numbers. This was motivated by the fact that the BCG 1.1 files generated by CADP tools on 64-bit architectures may now contain more than 2^{32} states and/or edges, so that certain quantities (state numbers, edge numbers, cardinality of state and edge sets, number of levels in breadth-first or depth-first search explorations, etc.) can now exceed the limits of XTL's 32-bit "integer" type.

A new predefined data type, named "number", has been added to the XTL language to allow the manipulation of state, label, and edge numbers, which were previously manipulated as values of type "integer". This new type is equipped with the usual arithmetic and relational operations and has a special syntax for constants. The operations handling state, label, and edge numbers (and also the cardinality of the sets thereof), which previously returned values of type "integer", were modified in order to return values of type "number". Several libraries containing various XTL operators were modified in order to use the "number" type instead of the "integer" type at appropriate places, and the XTL manual page was updated accordingly.

5.1.4. Compositional Verification Tools

Participants: Frédéric Lang, Radu Mateescu.

The CADP toolbox contains various tools dedicated to compositional verification, among which EXP.OPEN 2.0, PROJECTOR 3.0, BCG_MIN, 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 3.0 implements behavior abstraction [57], [65] by taking into account interface constraints. BCG_MIN minimizes behaviour graphs modulo strong or branching bisimulation and their stochastic extensions. SVL (*Script Verification Language*) is both a high level language for expressing complex verification scenarios and a compiler dedicated to this language.

In 2009, we corrected a few bugs in some of these tools and we enhanced them along the following lines:

- We added to SVL a new operator "**prio**", which allows priority to be given to some transitions over others. This operator already existed in EXP.OPEN 2.0, but was not available in SVL scripts. SVL implements the "**prio**" operator by calling the EXP.OPEN 2.0 tool. We also extended SVL with the possibility of generating EXP files containing hierarchical networks of communicating automata (i.e., which include other EXP files recursively).
- We continued the work initiated in 2008 about a technique, named CCD (*Compositional Confluence Detection*) that finds confluent transitions (not necessarily τ-transitions) in a network of communicating automata to identify transitions that are τ-confluent in the product graph. Following known results on τ-confluence, these transitions can be given priority in the product graph, thus allowing to generate a smaller graph while preserving branching equivalence. We also designed a variant of CCD that preserves deadlocks. CCD can be combined with other techniques, such as compositional verification, on-the-fly verification, and partial order reduction. We implemented CCD in EXP.OPEN 2.0 and made a series of experiments that showed its benefits on several examples. A paper on CCD was published in an international conference [28].

- We started to implement a new version of the BCG_MIN tool, using a (sequential) partition refinement algorithm based on state signatures [45]. We adapted this minimization algorithm to address also the stochastic extensions of strong and branching bisimulations. We validated our new version of BCG_MIN on a package consisting of 266 normal LTSs and 283 stochastic and probabilistic LTSs, as well as on the VLTS benchmark suite. As regards performance, strong and branching minimization of all the VLTS graphs using the new version of BCG_MIN takes 1.5 times less memory and 42 times less time than the previous version.

5.1.5. Performance Evaluation Tools

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

In addition to its verification capabilities, the CADP toolbox contains several tools dedicated to performance evaluation, namely BCG_MIN, BCG_STEADY, BCG_TRANSIENT, and DETERMINATOR. Contrary to most CADP tools that operate on labeled transition systems, these tools operate on probabilistic/stochastic models derived from discrete-time and continuous-time Markov chains.

In 2009, we added a new performance evaluation tool, named CUNCTATOR, which performs steady-state simulation of continuous-time Markov chains on the fly. CUNCTATOR takes as input a graph represented implicitly using the OPEN/CÆSAR environment, converts it to a continuous-time Markov chain by applying user-defined hiding and renaming operations, and explores a random execution sequence on the fly. During the exploration, the tool sums up the virtual time elapsed in the stable states (i.e., without outgoing τ-transitions) of the sequence, determined according to rates of the stochastic transitions going out of these states. The simulation terminates when either the virtual time, or the length of the simulation sequence (number of transitions) reaches a maximum value specified by the user. Upon termination, the tool displays the throughputs of the labeled stochastic transitions of interest, together with other statistical information about the simulation (number of τ-transitions encountered, presence of nondeterminism, etc.).

When a nondeterministic state (i.e., a state with at least two outgoing τ-transitions) is reached, CUNCTATOR advances the simulation sequence along one of the outgoing τ-transitions. The choice of this τ-transition can be currently made in three ways, each one corresponding to a different scheduler: (a) the first τ-transition encountered is chosen, (b) the last τ-transition encountered is chosen, or (c) a transition is chosen with equal probability among the τ-transitions going out of the current state. When a simulation has encountered nondeterministic states, the user has the possibility of launching other simulations using different schedulers. This allows to establish whether the Markov chain is nondeterministic (two simulations differing only in their schedulers yield different throughput values), which indicates that the stochastic model under investigation is semantically incorrect.

CUNCTATOR also allows to save the context reached at the end of a simulation in order to restart subsequent simulations from this context. This allows to achieve a linear-time complexity for a sequence of increasingly long simulations, each one being the prefix of the subsequent ones.

CUNCTATOR operates by storing in memory only the current state of the Markov chain (i.e., the last state of the simulation sequence), thus consuming only a small amount of memory, independent from the length of the simulation sequence and from the size of the Markov chain. Compared to BCG_STEADY, which computes the exact throughputs of labeled stochastic transitions of Markov chains represented explicitly as BCG files, CUNCTATOR consumes less memory but may require a longer execution time in order to compute the desired throughputs with the same accuracy as BCG_STEADY.

5.1.6. Other Tool Developments

Participants: Hubert Garavel, Yves Guerte, Rémi Hérilier, Frédéric Lang, Radu Mateescu, Sébastien Meriot, Jeanne Merle, Louis Paternault, Wendelin Serwe, Damien Thivolle, Anton Wijs.

A key objective for the future of CADP is the ability to support recent computing platforms. This is a heavy task because of the number of tools in CADP, their intrinsic complexity, and their reliance upon third-party software. In 2009, we continued our efforts in this direction:

- We brought support for two recent operating systems: WINDOWS 7 and MACOS 10.6 ("Snow Leopard").
- We enhanced all CADP tool to handle large files, beyond the limit of 2 or 4 Gbytes.
- We completed our collaboration with Pierre Boullier and Benoît Sagot (ALPAGE project-team of INRIA Rocquencourt) to port the SYNTAX compiler generation software (more than $300,000$ lines of C code) to 12 different platforms (processor, operating system, and C compiler), and especially to 64-bit architectures. In 2009, we solved together a few remaining bugs in the WINDOWS version of SYNTAX and we enhanced the tool to support large files.

Because of the growing usage of CADP in industry and academy, we pursued our efforts to master the software quality of CADP:

- We continued building a comprehensive validation framework, based on non-regression testing and semantical checking for the CADP tools. This framework allows functional testing of individual tools as well as integration testing for several CADP tools used altogether to perform complex verification scenarios on various computing platforms and using various compilers.
- We continued gathering large collections of benchmarks (BCG graphs, boolean equation systems, μ-calculus formulas, etc.) for testing the CADP tools extensively.
- For the development of CADP, we continued experimenting the PIPOL platform of INRIA, which provides reservation, configuration, and deployment facilities for porting and testing applications across various hardware and software environments.

We corrected three memory leaks related to the τ-compression and τ-confluence reductions. This change reduced the memory consumption of the BISIMULATOR, DISTRIBUTOR, EVALUATOR, and REDUCTOR tools when handling graphs containing many τ-transitions. We also corrected two bugs in the EVALUATOR model checker.

We undertook the study of several weak variants of τ-confluence [58], with the goal of increasing the reduction for graphs containing many sequences of τ-transitions. Intuitively, the weaker variants allow for more reduction (because they detect more confluent τ-transitions), but are computationally more expensive (because they require to compute transitive closures on τ-transitions). We proposed encoding schemes using alternation-free boolean equation systems for eight τ-confluence variants, ranging from strong to ultra weak τ-confluence. We also proposed a hierarchical encoding for each variant, which consists in detecting first the τ-transitions confluent w.r.t. the stronger variants of τ-confluence before trying the current variant itself [32]. We developed a prototype implementation of these encodings using the CÆSAR_SOLVE library and experimented it on graphs taken from the CADP demo examples and from the VLTS benchmark suite. These experiments allowed us to identify the most effective weak τ-confluence variants, which improved the amount of reduction up to one order of magnitude compared to strong τ-confluence, without increasing the computation time significantly.

We pursued our work, undertaken in 2008, on analyzing the properties of semantic WEB applications using verification tools. These applications have as underlying models directed graphs describing the resources involved and their properties. These graphs can be conveniently expressed in the RDF (*Resource Description Framework*) [64] semantic WEB language, which can be subsequently converted into the BCG file format by using the prototype RDF.OPEN tool in conjunction with GENERATOR. In 2009, we focused our attention on the analysis of RDF descriptions by means of CADP. To this purpose, we defined a translation from the SPARQL [74] query language used by the semantic WEB community to query RDF files, to the XTL language (see § 5.1.3). The translation is compositional, each operator of the SPARQL relational algebra being translated into an XTL iteration macro, which enumerates all the solutions corresponding to the evaluation of the operator on an RDF graph encoded as a BCG file [31]. We experimented this approach by translating typical examples of SPARQL queries into XTL manually, and evaluating them on several examples of RDF graphs encoded as BCG files. The results indicate that the evaluation of queries using XTL (not considering the time necessary for loading the BCG file in memory) has a performance comparable with that of dedicated SPARQL engines, such as ARQ [2].

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

- the CTTOOL tool for analyzing distributed Fractal components [41] developed at INRIA Sophia-Antipolis;
- the C.OPEN tool for model checking C programs with dynamic memory allocation [85] developed at the University of Málaga (Spain);
- the AVATR tool [75] for the automatic verification and analysis of test results of OCÉ printers, developed at the University of Twente (The Netherlands);
- a testing tool based on coverage criteria [83] developed at the Graz University of Technology (Austria);
- the VERCORS platform for the specification and verification of distributed FRACTAL/GCM components [49] developed at INRIA Sophia-Antipolis;
- the FLORA framework for the decomposition and implementation of software architectures for local recovery [81], [48], developed at the University of Twente (The Netherlands);
- the TTOOL for the DIPLODOCUS environment for SOC design space exploration [38], [56], [84] developed at the LABSOC laboratory of ENST (France);
- the DATALOG_SOLVE static analysis tool [37] developed at the University of Valencia (Spain);
- the CLOVE tool for validating CRESS composed grid services [82] developed at the University of Stirling (Scotland, United Kingdom);
- the CORAL tool [47] for analyzing availability of systems, developed at the University of Saarbrücken (Germany) and the University of Twente (The Netherlands);
- the ITACA-ACIDE integrated toolbox for the automatic composition and adaptation of Web services [52], [53] developed at the University of Málaga (Spain);
- a process algebraic performance evaluation framework [68] developed at the University of Twente (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 [3] for the data type part of LOTOS, the CÆSAR compiler [10] for the process part of LOTOS, and the CÆSAR.INDENT pretty-printer.

In 2009, in addition to fixing five bugs in the CÆSAR and CÆSAR.ADT compilers, we improved the LOTOS dedicated tools of CADP as follows:

- On 64-bit architectures, the maximum number of states that CÆSAR can generate directly and store has been increased from 2^{32} to 2^{34}.
- Similarly to the introduction of dynamic resizing of hash tables in OPEN/CÆSAR's tables (see § 5.1.2), the state table used by CÆSAR during graph generation has also been made extensible.
- CÆSAR now gathers information about the current machine architecture and the amount of physical memory available. With this information, CÆSAR now generates simulator programs that are optimized to store as many states as possible on the machine.
- The hash function used by CÆSAR has been improved. This leads to faster computation and better dispersion.

[2] http://jena.sourceforge.net/ARQ

5.2.2. Compilation of LOTOS NT

Participants: Hubert Garavel, Yves Guerte, Frédéric Lang, Christine McKinty, Wendelin Serwe, Gideon Smeding, Jan Stoecker.

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

- We continued enhancing our TRAIAN 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 [8].

  In 2009, we corrected four bugs and enhanced TRAIAN in several respects: we changed the syntax of the nondeterministic choice statement to make it compatible with the syntax accepted by the LNT2LOTOS translator, we made it easier to define the range of natural and integer values in LOTOS NT descriptions, and we adapted the C code generated for tester and selector operations to make it compatible with the libraries already present in CADP.
- In the context of the MULTIVAL contract (see § 6.2), we continued improving the LNT2LOTOS tool suite that translates (a large subset of) LOTOS NT into LOTOS, thus allowing the use of CADP to verify LOTOS NT descriptions. This tool suite is being used by BULL to model and verify critical parts of its FAME2 multiprocessor architecture (see § 5.3.1); it was also used to verify protocol specifications provided by AIRBUS (see § 5.3.5).

  In 2009, the LNT2LOTOS tool suite was enhanced significantly. Four successive versions were released, which corrected several bugs and brought many improvements, with the objective to prepare a future integration of this tool suite in CADP.

  As regards the data type part of LOTOS NT, we added support for floating-point numbers and for inclusion of modules; we enriched the set of predefined operations for boolean, character and string types; we compiled constructor operations for the integer type efficiently; we extended the syntax of hexadecimal and character constants; and we enhanced the "**case**" statements to handle constants and free variables.

  As regards the process part of LOTOS NT, we improved the translation of parallel composition of processes to support the "**disrupt**" and "**break**" statements inside parallel compositions; we simplified the syntax of "**loop**" statements and the code generated for the "**case**" and "**raise**" statements; we extended the syntax of patterns with expressions containing non-constructor operations; we simplified the LOTOS code generated for exit handling; we added generalized support for constructor operations; and we improved the translation of nested compound and "**if-then-else**" statements in order to reduce the computation effort and the size of the generated LOTOS code.

  New options were added to select a specific LOTOS NT process as being the main process of the resulting LOTOS specification. The management of intermediate files was simplified, and error handling was made compatible with the CÆSAR and CÆSAR.ADT compilers for LOTOS.

  A new OPEN/CÆSAR-compliant compiler named LNT.OPEN was introduced to explore the state spaces of LOTOS NT descriptions on the fly.

  LNT2LOTOS is developed using the SYNTAX/TRAIAN technology. Currently, it represents $35,900$ lines of code ($4,000$ lines of SYNTAX code, $29,000$ lines of LOTOS NT code, and $2,900$ lines of C code). The LOTOS NT reference manual [35] was updated and grew from 71 pages (at the end of 2008) to 101 pages. The non-regression database was enriched with about $1,160$ new LOTOS NT programs.
- As regards the compilation of temporal extensions, we published in an international conference a paper [30] about the new intermediate form ATLANTIF [15], into which languages combining data, concurrency, and real-time (such as E-LOTOS and LOTOS NT) could be translated efficiently. ATLANTIF is based upon the NTIF model [7] 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: Hubert Garavel, Claude Helmstetter, Rémi Hérilier, Frédéric Lang, Wendelin Serwe, Gideon Smeding, Damien Thivolle.

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

In 2009, besides the LNT2LOTOS tool suite (see § 5.2.2), we worked on the following translators:

- In the framework of the OPENEMBEDD (see § 6.3) and TOPCASED (see § 6.4) projects, and in cooperation with the LAAS-CNRS and IRIT laboratories, we continued the development of tools for FIACRE (*Format Intermédiaire pour les Architectures de Composants Répartis Embarqués*). Derived from NTIF [7] and VCOTRE [44], FIACRE is a pivot formalism between modeling languages (such as AADL, UML, or SYSML) and verification tools (such as CADP and TINA). FLAC (*Fiacre to Lotos Adaptation Component*) is a tool that translates FIACRE (except priorities and time constraints) into a behaviourally equivalent LOTOS specification and an auxiliary SVL script.

  In 2009, we used FLAC to verify formally the specification of an ATC (*Air Traffic Control*) system provided by AIRBUS (see § 5.3.5). This case study enabled to correct 13 bugs in the LOTOS code generated by FLAC, leading to version 1.3 of FLAC.

- We considered the process algebra FSP (*Finite State Processes*) defined in a popular textbook on concurrency [67] and supported by the LTSA tool designed at Imperial College (London, United Kingdom). For this language, we developed the FSP2LOTOS tool, which translates FSP into LOTOS, EXP.OPEN 2.0, and SVL code [77].

  In 2009, we finalised the development of the FSP2LOTOS translator. We implemented the operator "#" of FSP, which returns the number of elements in a set of labels, and the operator "**forall**" in composite processes, which implements parallel composition between a set of processes indexed by a set of labels.

  We also developed a new tool, named FSP.OPEN, which provides a transparent interface between FSP and the OPEN/CÆSAR environment. This tool first invokes FSP2LOTOS and then EXP.OPEN 2.0 on the generated network of LTSs.

  We enriched our non-regression test database with 72 new examples of FSP programs provided by Charles Pecheur (Université Catholique de Louvain). These examples revealed a bug in FSP2LOTOS, which we corrected.

  An updated article on FSP2LOTOS was accepted for publication in an international journal [18].

- In the context of the INRIA/LETI collaboration (see § 7.1) and the MULTIVAL contract (see § 6.2), 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 [61] 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.

  Our prior work led to two approaches for translating the PV (*Programmers View*) level of SYSTEMC/TLM (i.e., TLM models without explicit timing information) into LOTOS:

  – Our first approach [12] follows the "official" simulation semantics of SYSTEMC, keeping the scheduler as defined in [61].

  - Our second approach [13] proposes a fully asynchronous semantics for TLM, getting rid of the central scheduler that is difficult to implement in parallel circuits and prevents certain execution sequences from being analyzed during formal verification.

  In 2009, we applied this approach to a large industrial case study, the *Blitter Display* (see § 5.3.4).

  We also developed a new approach that directly connects TLM to the OPEN/CÆSAR environment, avoiding any intermediate translation into LOTOS. This led to a new OPEN/CÆSAR-compliant compiler, named TLM.OPEN, which reuses the G++ compiler and the OSCI SYSTEMC and TLM libraries. The numerous tools of CADP can be used to verify complex properties and check the equivalence of two TLM programs. The advantages of the new approach are the following:

  - TLM.OPEN is much simpler: engineers do not need to learn LOTOS and they have fewer lines of code to write (typically, additional functions for storing and restoring the current state of each TLM module, i.e., generally less than 20% of the size of the TLM model).
  - TLM.OPEN allows the verification of larger TLM models. In [12], we verified a benchmark containing m interrupt transmitter modules up to $m = 19$; TLM.OPEN can handle this benchmark up to $m = 22$ on the same machine. In [73], another benchmark containing n transmitters is described and verified up to $n = 13$ using the LUSSY [72] and SMV [69] tools; TLM.OPEN can handle this benchmark up to $n = 18$ using the same amount of memory.

- BPEL (*Business Process Execution Language*) [63] is a language inspired by the π-calculus [71] and standardized by the OASIS consortium (led by IBM and Microsoft) to describe the orchestrations of Web services. BPEL depends on other XML-related languages such as XML Schema, XPATH, and WSDL, a W3C standard to describe the interfaces of Web services.

  In 2009, we were contacted by two research groups that expressed their interest in having a translation from BPEL to LOTOS NT: the team of Pr. Stuart Madnick at MIT (who was interested in using LOTOS NT to resolve semantic incompatibilities in Web service composition) and the team of Pr. Valentin Cristea at the Polytechnic University of Bucharest (who was interested in verifying formally a BPEL case study using CADP).

  Following these requests, we defined a set of rules to translate into LOTOS NT a large subset of BPEL that we found to be used in practice. We applied these rules to translate manually in LOTOS NT several Web services provided to us by Pr. Madnick's team, which enabled MIT to continue its research work on the topic. This led to a publication [66] where our contribution is acknowledged.

  We also undertook the implementation of an automated translator from BPEL and WSDL into LOTOS NT. Currently, the translation of data types is finished and the translation of processes is under way.

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 § 6.2) contract, we studied together with BULL 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 2009, we pursued the study of the "*barrier*" primitive of MPI, which allows several parallel processes to synchronize (each process arriving at the barrier waits for all the others to arrive) before continuing their execution. The latency of the barrier primitive corresponds to the (average) time taken by a process to traverse the barrier, i.e., the time between the moment when it arrives and when it leaves the barrier. Our goal was to estimate this latency on different topologies, different software implementations of the MPI primitives, and different cache coherency protocols. We specified, using the LOTOS and C languages, five protocols implementing the barrier primitive (*centralized*, *combining*, *tournament*, *dissemination*, and *tree-based*), which differ by the shared data structures and the synchronizations used. We extended the LOTOS specifications of these five protocols with Markov delays corresponding to the read/write and fetch-and-decrement operations performed by these protocols.

In order to fight state explosion, we undertook the analysis of the Markov chains underlying these protocols by means of steady-state simulation, carried out on the fly using the newly developed CUNCTATOR tool (see § 5.1.5). CUNCTATOR enables to terminate a simulation when a certain virtual time is reached and does not attempt to estimate by itself the accuracy of the simulation results (this estimation may depend on the type of application under study). To analyze the barrier protocols, we considered two widely-used estimation methods [62] enabling to decide whether the results of a simulation are sufficiently close to their real values: the confidence interval method, which involves launching increasingly longer simulations, and the concurrent estimation method, which involves launching in parallel several simulations with different random parameters. We automated these two convergence methods by means of shell scripts and applied them to the five barrier protocols.

Using the confidence interval method, which relies upon the save/restore mechanism implemented by CUNCTATOR, we were able to compute the latencies of barrier traversals for all the five protocols, for configurations containing up to four processes. For the centralized and tree-based protocols (which could also be analyzed by completely generating their Markov chains and applying the BCG_STEADY tool), the throughputs computed by simulation were close (less than 5%) to those computed by BCG_STEADY. These results suggest that the latencies computed by simulation for the three other protocols with four processes (which are too complex to be analyzed by generating their Markov chains explicitly) are also close to their real values.

5.3.2. The FAUST/MAGALI Architectures

Participants: Radu Mateescu, Wendelin Serwe.

In the context of the INRIA/LETI collaboration (see § 7.1) and the MULTIVAL contract (see § 6.2), we pursued the study (started in 2005) of FAUST (*Flexible Architecture of Unified System for Telecom*), a platform based on NOC (*Network on Chip*) for wireless telecom applications (4G, MIMO, etc.), developed by the CEA/LETI laboratory [42]. Since 2007, we are focusing on the latest version, named MAGALI, of this architecture.

Together with LETI scientists (Francois Bertrand, Virang Shah, and Yvain Thonnart), we studied the communication interconnect, which routes packets (consisting of several 34-bit flits) between the 23 components of the 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 consists 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.

In 2009, LETI scientists (Virang Shah and Yvain Thonnart) developed a set of tools for the verification of netlists for asynchronous circuits. These tools automatically translate netlists and standard cells of a commercial cell library commercialized by STMICROELECTRONICS into LOTOS. These tools were applied, in conjunction with CADP, for analyzing a speed-optimized version of a component of the communication interconnect. Using the EVALUATOR model checker, a potential deadlock was exhibited, which had not been found using simulation. Because the deadlock only occurs under particular assumptions on differences of delay, LETI scientists are currently investigating the probability that these assumptions are satisfied in a real chip, to decide whether the optimized component should be integrated in the design.

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 § 6.2) 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 using 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 2009, the verification of the XSTREAM architecture benefited from the latest enhancements brought to the CADP toolbox. For instance, the enhancements of the OPEN/CÆSAR table (see § 5.1.2) significantly decreased the state space generation time. Also, the extended capabilities offered by the enhanced OPEN/CÆSAR table and by version 1.1 of the BCG format (see § 5.1.1) enabled the generation of very large transition systems for the XSTREAM architecture (up to 15 billion transitions).

We also continued performance evaluation studies to predict latency and throughput of communication between XSTREAM queues. We consolidated our IPC (*Interactive Probabilistic Chains*) approach undertaken in 2008. This approach is inspired by Interactive Markov Chains [60], but uses probabilities instead of stochastic distributions and a central clock governing all delays.

The semantic definition of latency was generalized, the IPC tool chain has been enhanced, and a conference paper was published [23], which presents the IPC approach and its application to XSTREAM.

5.3.4. The Blitter Display

Participants: Hubert Garavel, Claude Helmstetter, Wendelin Serwe.

In the context of the MULTIVAL contract (see § 6.2), we started to study whether the CADP tools could enrich with formal verification capabilities the current design flow of STMICROELECTRONICS, which is based on SYSTEMC/TLM. To this aim, STMICROELECTRONICS provided us with the *Blitter Display* (BDISP for short), a 2D-graphics co-processor implementing BLIT (*Block Image Transfer*) and numerous graphical operators, e.g., rotations, alpha blending, or Blue Ray disc decoding. The BDISP is software-controlled through instructions written in nodes of up to four application queues and two real-time composition queues. It is described by more than $25,000$ lines of TLM code.

In 2009, we continued the verification of the BDISP following both approaches mentioned above (see § 5.2.3):

- We improved the translation scheme from TLM to LOTOS, optimized the hand-written LOTOS code (reducing the size of a state by a factor of 10), and ported the model to 64-bit architectures (so as to use more than 4 Gb of memory). Although these improvements did still not allow to generate the complete state space, we were able to generate bigger graphs and to model check more correctness properties using the EVALUATOR tool than in 2008.

  We discovered a synchronization issue that could be reproduced using an interactive SYSTEMC scheduler on the untimed version of the TLM model (i.e., replacing delayed notifications by immediate notifications), but not on the timed version (nor on the actual circuit itself). We proposed a modification of the TLM model so that the property holds also in an untimed context. This work led to a publication in an international conference [25].

- We developed a new TLM model of only the control part of the BDISP based on the specification documents. Experimenting the TLM.OPEN tool (see § 5.2.3) on this new TLM model, we were able to generate a complete state space of less than $10,000$ states, without any translation into LOTOS. These results have been presented to STMICROELECTRONICS, which showed interest in experimenting with TLM.OPEN on a new case study.

5.3.5. The Airbus Avionics Case Studies

Participants: Simon Bouland, Hubert Garavel, Wendelin Serwe, Damien Thivolle.

In the context of the TOPCASED project (see § 6.4), we have been studying how CADP can be used to verify avionics protocols. We addressed three successive case-studies provided to us by AIRBUS:

- We continued our study undertaken in 2008 of a ground/plane communication protocol based on TFTP (*Trivial File Transfer Protocol*). This protocol was specified as an automaton described in SAM [14], a graphical synchronous language developed by AIRBUS.

  To verify this protocol using CADP, a LOTOS NT specification was produced for an entire system in which two synchronous TFTP automata execute asynchronously and communicate with each other using UDP (*User Datagram Protocol*) links. This is a typical example of a GALS (*Globally Asynchronous, Locally Synchronous*) system. The TFTP automata were produced using a systematical translation from SAM to LOTOS NT. The UDP links, which may lose, duplicate and/or reorder messages, were modelled as bounded FIFO queues and bag data structures. Then, 29 correctness properties were specified in temporal logic and verified using the EVALUATOR 3.5 and 4.0 model checkers of CADP. This revealed 19 errors; simulations were carried out using the EXECUTOR tool of CADP to quantify the impact of these errors on the overall protocol performance.

  In 2009, we enhanced the approach in several respects. We optimized the LOTOS NT code to reduce the generated state spaces. We developed SVL scripts to automate the approach. Using the 64-bit versions of CADP, we generated larger BCG graphs (up to 1.5 billion states and 6.8 billion transitions). We also enhanced the model checking approach by hiding, in those BCG graphs, all labels that are not relevant for a given temporal logic formula. These results were presented during the COMPASS'09 workshop and led to a publication in an international conference [26]. An extended report about this work was written.

- Following the success of the TFTP case study, AIRBUS provided us with another example, the BITE (*Built In Test Equipment*)/CMS (*Central Maintenance Function*), which serves to manage equipment failure on A340 and A380 airplanes.

  We applied to this example the same methodology we developed for the TFTP. The two SAM automata modelling the BITE kernel and the CMS were translated into a set of LOTOS NT types and functions with boolean inputs/outputs (about $2,000$ lines of LOTOS NT code). Compared to the TFTP, this step was made fully automatic by developing an ACCELEO plugin that implements this translation within ECLIPSE.

  Additional LOTOS NT code (about $2,000$ lines) was written manually to model: (1) the two "wrapper" processes encapsulating the LOTOS NT functions generated from the SAM automata; (2) extra BITE protocol functionality not specified in the original SAM automaton; (3) bounded FIFO queues connecting the different parts of the BITE/CMS system; and (4) the expected behaviour of the system's environment.

  Then, the LOTOS NT specification was analyzed using the CADP tools: step-by-step simulation, generation of state spaces (about $100,000$ states for a FIFO of length 3), equivalence checking and model checking of various properties expressed in temporal logic regarding, e.g., the activation/expiration of timers and the correct restransmission of requests on timeouts.

- We finally addressed a third case study provided by AIRBUS, the ATC (*Air Traffic Control*) system, which had already been investigated in the framework of the European ITEA project SPICES.

  The ATC system was originally specified in AADL and then converted automatically into a FIACRE model using the AADL to FIACRE translator developed in the TOPCASED project. By studying the generated FIACRE model carefully, we discovered that the translation of shared AADL variables was not optimal. We therefore improved manually the FIACRE model, replacing certain shared variables by constants, and ensuring that each remaining shared AADL variables is translated into one single shared FIACRE variable — instead of several ones (between two and four) in the generated FIACRE code.

We then applied the FLAC translator (see § 5.2.3) to convert the FIACRE model (about $1,000$ lines) into LOTOS (six concurrent processes, about $3,400$ lines of code). This revealed 13 bugs in the FLAC translator, that were corrected by X. Clerc (INRIA Paris-Rocquencourt) and F. Lang.

The ATC system heavily relies on priorities and quantitative time constraints, which cannot be directly expressed in LOTOS and are not handled by the FLAC translator. Therefore, we introduced at three places a new LOTOS action ("tick") that models the elapse of time. To ensure that this "tick" action has a lower priority than all other actions, and more generally to implement the concept of priorities that exist in FIACRE, we developed an enhanced version of the GENERATOR tool of CADP.

Using CADP, we generated the corresponding state space (about $308,000$ states and $421,000$ transitions) and proved that it was branching equivalent with the state space obtained by applying the FRAC and TINA tools of LAAS-CNRS to the original FIACRE model. This cross-verification confirmed the absence of deadlock property, which had already been shown using TINA. Using the EVALUATOR model checker of CADP, we also verified that the AADL scheduler correctly executes all ATC processes during each time period.

We then focused verification on concurrent accesses to shared AADL variables, an analysis that cannot be done in TINA as FRAC does not generate any action when a shared variable is accessed. To our surprise, we discovered two distinct race condition problems (destructive concurrent writes on a shared variable), one during the initialization phase (during the first period, after 8 actions) and another one that occurs much later (around the 10th period, after 333 actions). In both cases, after a transitory phase, the ATC system enters into an infinite loop (livelock) in which all shared variables are no longer accessed, except for being periodically reset to zero, and all concurrent processes continue to execute but do not perform any useful task.

5.3.6. 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 performance analysis of stimulus rich reactive interfaces [79];
- the formal modeling and performance evaluation of gossiping networks [51];
- the verification of TOOLBUS scripts [55];
- the performance analysis of a train odometer controller [46];
- the realizability checking of collaboration diagrams [78];
- the verification of a turntable system [70];
- the formal analysis of consensus protocols in asynchronous distributed systems [39];
- the dependability and survivability evaluation of a water distribution process [76];
- the verification of semantic composability in simulation models [80];
- the formal specification and analysis of accelerated heartbeat protocols [40].

6. Contracts and Grants with Industry

6.1. The EC-MOAN Project

Participants: Hubert Garavel, Radu Mateescu, 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 IBIS 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 2009, our efforts focused on developing new algorithms that improve the performance of state space generation using caching techniques [29]. We also studied and implemented on the fly reductions based on several variants of weak τ-confluence [32], a form of partial order reduction preserving branching equivalence between graphs. We experimented the effect of these reductions on graphs modeling the behaviour of *Escherichia coli* [33], obtained by qualitative simulation of the corresponding genetic regulatory networks using the GNA (*Genetic Network Analyzer*) tool developed by the IBIS project-team. Finally, we contributed to the design of a connection between GNA and CADP based on Web services [19].

6.2. The Multival Project

Participants: Nicolas Coste, Hubert Garavel, Yves Guerte, Rémi Hérilier, Holger Hermanns, Frédéric Lang, Etienne Lantreibecq, Radu Mateescu, Louis Paternault, Wendelin Serwe, 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 Unique Interministériel*) 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, and was extended for a fourth year in December 2009. In 2009, we focused our activities on the enhancement of CADP (see § 5.2.2 and § 5.2.3) and case studies in collaboration with our partners to verify and predict the performance of the architectures FAME2 (see § 5.3.1), FAUST/MAGALI (see § 5.3.2), and XSTREAM (see § 5.3.3).

6.3. The OpenEmbedd Project

Participants: 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 2009, we continued the development of our automated translator from the FIACRE asynchronous intermediate model for embedded systems [43] to LOTOS (see § 5.2.3). The translator was integrated as a plugin in the OPENEMBEDD platform. A paper about the results of OPENEMBEDD was published in a national conference [21].

The OPENEMBEDD project ended on April 30, 2009. It was evaluated positively by ANR (*Agence Nationale de la Recherche*), after a final meeting that took place in Rennes on March 26–27, 2009.

6.4. The Topcased Project

Participants: Simon Bouland, Hubert Garavel, Frédéric Lang, Jeanne Merle, Wendelin Serwe, 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 (*Computer-Aided Software Engineering*) environment providing methods and tools for embedded system development, ranging from system and architecture specifications to software and hardware implementation through equipment definition. VASY contributes to the combination of model-driven engineering and formal methods for asynchronous systems.

TOPCASED started in August 2006 for 40 months. In 2009, we enhanced the FLAC translator from FIACRE to LOTOS (see § 5.2.3) and we worked on three case studies provided to us by AIRBUS (see § 5.3.5).

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

From 2004 to 2009, the VASY project-team played an active role in the joint research center 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 developed 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 2009, this collaboration was pursued as part of the MULTIVAL project (see § 6.2).

Additionally, we collaborated in 2009 with several INRIA project-teams:

- ATLAS (Nantes): collaboration in the framework of the OPENEMBEDD national action (Jean Bézivin and Frédéric Jouault);
- ALPAGE (Rocquencourt): enhancements to the SYNTAX V6 compiler generation software (Pierre Boullier, and Benoît Sagot);
- ESPRESSO (Rennes): collaboration in the framework of the TOPCASED and OPENEMBEDD national actions (Jean-Pierre Talpin and Julio Peralta);
- IBIS (Rhône-Alpes): applications of model checking to biological systems (Estelle Dumas, Hidde de Jong, Pedro Monteiro, and Michel Page).

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

- Charles Pecheur at Université Catholique de Louvain (Louvain-la-Neuve, Belgium).
- LAAS-CNRS laboratory (Toulouse): collaboration in the framework of the OPENEMBEDD and TOPCASED projects (Bernard Berthomieu and François Vernadat);
- IRIT laboratory (Toulouse): collaboration in the framework of the TOPCASED project (Mamoun Filali and Jean-Paul Bodeveix).

7.2. International Collaborations

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.

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

In addition to our partners in aforementioned contractual collaborations, we had scientific relations in 2009 with several international universities and research centers, including:

- Imperial College (Jeff Kramer and Jeff Magee),
- LIAMA, China (Claude Helmstetter and Vania Joloboff),
- Massachusetts Institute of Technology (Stuart Madnick and Harry Zhu),
- Polytechnic University of Bucharest (Valentin Cristea),
- Saarland University (Pepijn Crouzen, Holger Hermanns, Sven Johr, and Reza Pulungan),
- Université de Sherbrooke (Romain Chossart, Marc Frappier, and Benoît Fraikin),
- Université Catholique de Louvain (Charles Pecheur and José Vander Meulen),
- University of Málaga (Gwen Salaün),
- University of Malta (Gordon Pace and Sandro Spina), and
- University of Twente (Stefan Blom and Jaco van de Pol).

7.3. Visits and Invitations

In 2009, we had the following scientific exchanges:

- R. Mateescu, H. Garavel, and M. Zidouni visited the Dependable Systems and Software group of Saarland University (Saarbruecken, Germany) on February 3–5, 2009.
- Charles Pecheur (professor at Université Catholique de Louvain, Belgium) visited us on February 13, 2009.
- Doron Peled (professor at the Bar Ilan University, Israel) visited us on February 18, 2009.
- On March 23, 2009, VASY hosted a delegation of fifteen professors from various Chinese universities and research institutes. H. Garavel presented the current status of the VASY project-team. W. Serwe gave a demonstration of the CADP toolbox, in particular LOTOS and the EXEC/CÆSAR framework. F. Lang gave a talk entitled "*Refined Interfaces for Compositional Verification*". R. Mateescu gave a talk entitled "*Distributor and Bcg_Merge: Tools for Distributed Explicit State Space Generation*" and a demonstration of the DISTRIBUTOR and BCG_MERGE tools of CADP.
- S. Bouland visited Patrick Farail and Pierre Gaufillet (Software Engineering Methods, AIRBUS, Toulouse) on March 12–17, 2009 and August 2–10, 2009.
- Vania Joloboff (LIAMA laboratory, China) visited us on April 10, 2009.
- Sylvain Rampacek (lecturer at the IUT Dijon) visited us on April 20–24, 2009.
- Stefan Blom (University of Twente, The Netherlands) visited us on May 19, 2009.
- Jaco van de Pol (University of Twente, The Netherlands) visited us on June 30, 2009.
- G. Salaün visited the University of Málaga (Spain) on November 17–21, 2009, as supervisor of Javier Cámara's PhD thesis, which was defended on November 18, 2009.

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 2009, the main facts are the following:

- We prepared and distributed 10 successive beta-versions (from 2007-p to 2007-r and from 2008-a to 2008-g "Zurich") of CADP.
- The number of license contracts signed for CADP increased from 394 to 411.
- We were requested to grant CADP licenses for 760 different computers in the world.
- The TRAIAN compiler was downloaded by 35 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 2009, this forum had 118 registered users and 705 messages exchanged.

Since June 3, 2009, we created a Wikipedia page for CADP.

8.2. Program Committees

In 2009, the members of VASY took on the following responsibilities:

- F. Lang was a program committee member of IFM'2009 (*7th International Conference on Integrated Formal Methods*), Düsseldorf, Germany, February 16–20, 2009.
- R. Mateescu was a program committee member of TACAS'2009 (*15th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*), York, United Kingdom, March 22–29, 2009.
- F. Lang was a program committee member of NEPTUNE'2009 (*Nice Environment with a Process and Tools Using Norms and Example*), Paris, France, May 26–27, 2009.
- F. Lang was a program committee member of WICSA/ECSA'2009 (*Joint Working International Conference on Software Architecture and European Conference on Software Architecture*), Cambridge, United Kingdom, September 14–17, 2009.
- G. Salaün was a program committee member of WASELF'2009 (*2nd Workshop on Autonomic and SELF-adaptive Systems*), San Sebastián, Spain, September 8, 2009.
- H. Garavel and H. Hermanns were program committee members of FMICS'09 (*14th International Workshop on Formal Methods for Industrial Critical Systems*), Eindhoven, The Netherlands, November 2–3, 2009.
- R. Mateescu was a program committee member of PDMC'2009 (*8th International Workshop on Parallel and Distributed Methods in verifiCation*), Eindhoven, The Netherlands, November 4, 2009.

8.3. Lectures and Invited Conferences

In 2009, we gave talks in several international conferences and workshops (see bibliography below). Additionally:

- R. Mateescu and A. Wijs participated to the 5th EC-MOAN meeting held at CWI (Amsterdam, The Netherlands) on January 12–13, 2009. A. Wijs gave a talk entitled "*On-the-Fly Reduction of Automata based on Weak Tau-Confluence*" on January 12, 2009.
- W. Serwe and C. Helmstetter participated to the "*Journées Informatique Massivement Multiprocesseur et Multicoeur*" held at INRIA Rocquencourt (France) on February 4–5, 2009. W. Serwe gave a talk entitled "*Verification and Performance Evaluation Tools for Multiprocessor Architectures*" on February 4, 2009.
- H. Garavel participated to the 2nd winter school on "*Hot Topics in Distributed Computing*" held at La Plagne (France) on March 15–20, 2009.
- F. Lang participated to the final OPENEMBEDD meeting (ANR evaluation) held at INRIA Rennes (France) on March 26–27, 2009. He demonstrated the FIACRE to LOTOS translator FLAC and its application to the verification of a FIACRE program using CADP.
- H. Garavel participated to the COMPASS'2009 workshop held at York (United Kingdom) on March 29, 2009. He gave an invited talk entitled "*Verification of Synchronous/Asynchronous Systems by Combining Model-Driven Engineering and Formal Methods*" (jointly written with D. Thivolle).
- H. Garavel, Y. Guerte, R. Hérilier, C. Helmstetter, R. Mateescu, and W. Serwe participated to the 9th MULTIVAL quarterly meeting held at INRIA Grenoble (France) on April 2, 2009. C. Helmstetter gave two talks, entitled "*Un modèle* LOTOS *du Blitter*" and "TLM.OPEN*: connecting* TLM *to* CADP".
- C. Helmstetter gave an invited presentation entitled "SYSTEMC.OPEN*: connecting* TLM *to* CADP" at INRIA Lille (France) on May 6, 2009. He also gave an overview of the SIMSOC project of the FORME project-team of the LIAMA laboratory, and discussed the generation of SYSTEMC/TLM models for simulation.

- R. Mateescu represented INRIA at the "ERCIM *Spring Days*" meeting held in Paris (France) on May 26–29, 2009. He gave a talk entitled "*Specification and Analysis of Biological Systems*" on May 27, 2009.
- C. Helmstetter gave a talk entitled "*Model Checking of* SYSTEMC/TLM *using the* CADP *Toolbox*" at STMICROELECTRONICS Grenoble (France) on June 16, 2009.
- R. Mateescu and A. Wijs participated to the 6th EC-MOAN meeting held in Edinburgh (United Kingdom) on June 16–17, 2009. A. Wijs gave a talk entitled "*On-the-Fly Property-Dependent Reduction of Automata*" on June 16, 2009.
- H. Garavel, C. Helmstetter, R. Mateescu, W. Serwe, and G. Smeding participated to the 10th MULTIVAL quarterly meeting held at CEA/LETI (Grenoble, France) on June 25, 2009. H. Garavel gave a talk entitled "*Développements récents de* CADP *mars-juin 2009*". C. Helmstetter gave a talk entitled "*Model Checking of* SYSTEMC/TLM *Models using the* CADP *Toolbox*". G. Smeding gave a talk entitled "*Announcing* LNT2LOTOS *4i*".
- R. Mateescu participated to the 7th EC-MOAN meeting held at Warwick (United Kingdom) on September 24, 2009. He gave a talk entitled "*Distributed Weak Tau-Confluence Reduction of Automata*".
- H. Garavel, F. Lang, R. Mateescu, and W. Serwe participated to the 11th MULTIVAL quarterly meeting held at STMICROELECTRONICS (Montrouge, France) on October 1st, 2009. H. Garavel gave a talk entitled "*Bilan des trois premières années de* MULTIVAL".
- C. Helmstetter attended the SBDCES'09 workshop held at Osaka (Japan) on October 7th, 2009. He gave a talk entitled "TLM.OPEN*: a* SYSTEMC/TLM *Front-End for the* CADP *Verification Toolbox*".
- H. Garavel attended the "*Embedded Systems Exhibition"* held in Grenoble (France) on October 14–15, 2009. He gave a talk entitled "*Asynchrony in Embedded Systems: The* MULTIVAL *Project*".
- F. Lang demonstrated the CADP toolbox at the 16th International Symposium on Formal Methods FM'2009 held in Eindhoven (The Netherlands) on November 2–6, 2009.
- G. Salaün participated to the annual seminar of the POPART project-team held in Corps (France) on November 5–6, 2009. He gave an invited talk entitled "*Composition of Service Protocols — Some Open Issues*" on November 6, 2009.
- C. Helmstetter gave a talk entitled "TLM.OPEN*: A* SYSTEMC/TLM *Front-End for the* CADP *Verification Toolbox*" at STMICROELECTRONICS (Grenoble, France) on November 12, 2009.
- H. Garavel gave an invited talk entitled "*The* TOPCASED *project*" at the conference "*Free Open Source Academia*" FOSSA held in Grenoble (France) on November 17–18, 2009.
- R. Mateescu gave a talk entitled "*Analyse de systèmes de transitions au moyen de systèmes d'équations booléennes*" at the LINA laboratory (Nantes, France) on November 19, 2009.
- H. Garavel, H. Hermanns, and R. Mateescu participated to a meeting held at BULL (Les Clayes sous Bois, France) on November 20, 2009. H. Garavel gave two talks, entitled "*Overview of* CADP" and "*Performance Evaluation of a* BULL SCSI-2 *Controller*". H. Hermanns gave a talk entitled "*A Compositional Approach to Perfomance Evaluation of Complex Systems*". R. Mateescu gave a talk entitled "*Analyzing Queuing Networks using* CADP".
- G. Salaün participated to the annual seminar of the SARDES project-team held in Autrans (France) on December 14–16, 2009. He gave two invited talks, entitled "*Composition of Service Protocols — Some Open Issues*" and "*The* CADP *Toolbox*", on December 15, 2009.
- H. Garavel was the invited speaker at the annual meeting of the "*FNRS Contact Group on Fundamental Computer Science*" held at Université Catholique de Louvain (Louvain-la-Neuve, Belgium) on December 16, 2009. He gave a talk entitled "*An Overview of* CADP *2009*".
- A. Kaufmann, R. Mateescu, and W. Serwe participated to the 12th MULTIVAL quarterly meeting held at STMICROELECTRONICS (Grenoble, France) on December 17, 2009. R. Mateescu gave a talk entitled "CUNCTATOR*: An On-the-Fly Simulator for Continuous-Time Markov Chains*". W. Serwe gave a talk entitled "LOTOS NT *to* LOTOS *version 4K: What is New in* LNT2LOTOS *&* LPP*?*".

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 Grenoble INP and Université Joseph Fourier.

In 2009:

- H. Garavel, F. Lang, and W. Serwe gave, jointly with Pascal Raymond (CNRS, Verimag), a course on "*Méthodes formelles de développement*" to the computer science engineering students of CNAM (*Conservatoire National des Arts et Métiers*) Grenoble (21 hours).
- F. Lang and W. Serwe gave the course on "*Temps réel*" to the 3rd year students of ENSIMAG (18 hours).
- H. Garavel was a jury member of Romain Bernard's PhD thesis entitled "*Analyses de sûreté de fonctionnement multi-systèmes*", defended at Université de Bordeaux I, on November 23, 2009.
- G. Salaün gave lectures on "*Algorithmics and Object-Oriented Programming*" to the 2nd year students and on "*Specification and Validation of Distributed Processes*" to the 3rd year students of ENSIMAG (64 hours).
- G. Salaün supervised the PhD thesis of Meriem Ouederni (University of Málaga, Spain), starting October 2008.
- G. Salaün supervised the PhD thesis of Javier Cámara (University of Málaga, Spain), defended on November 18, 2009.
- F. Lang and H. Garavel supervised the PhD thesis of Jan Stoecker (Grenoble INP), defended on December 10, 2009.

8.5. Miscellaneous Activities

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

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

H. Garavel is a member of the "*commission des ressources humaines*" of INRIA Grenoble Rhône-Alpes.

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

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

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. *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, vol. 209, Elsevier Science Publishers, April 2008, p. 149–164, http://hal.inria.fr/inria-00191141, Also available as INRIA Research Report RR-6368.

[3] H. GARAVEL. *Compilation of LOTOS Abstract Data Types*, in "Proceedings of the 2nd International Conference on Formal Description Techniques FORTE'89 (Vancouver B.C., Canada)", S. T. VUONG (editor), North-Holland, December 1989, p. 147–162.

[4] H. GARAVEL. *OPEN/CÆSAR: An Open Software Architecture for Verification, Simulation, and Testing*, in "Proceedings of the First International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS'98 (Lisbon, Portugal), Berlin", B. STEFFEN (editor), Lecture Notes in Computer Science, vol. 1384, Springer Verlag, March 1998, p. 68–84, http://hal.inria.fr/inria-00073337, Full version available as Inria Research Report RR-3352.

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

[6] H. GARAVEL, F. LANG. *SVL: a Scripting Language for Compositional Verification*, in "Proceedings of the 21st IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems FORTE'2001 (Cheju Island, Korea)", M. KIM, B. CHIN, S. KANG, D. LEE (editors), Kluwer Academic Publishers, IFIP, August 2001, p. 377–392, http://hal.inria.fr/inria-00072396, Full version available as Inria Research Report RR-4223.

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

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

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

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

[12] C. HELMSTETTER, O. PONSINI. *A Comparison of Two SystemC/TLM Semantics for Formal Verification*, in "Proceedings of the 6th ACM-IEEE International Conference on Formal Methods and Models for Codesign

MEMOCODE'2008 (Anaheim, CA, USA)", IEEE Computer Society Press, June 2008, p. 59–68, http://hal.inria.fr/inria-00275456.

[13] O. PONSINI, W. SERWE. *A Schedulerless Semantics of TLM Models Written in SystemC via Translation into LOTOS*, in "Proceedings of the 15th International Symposium on Formal Methods FM'08 (Turku, Finland)", J. CUELLAR, T. MAIBAUM, K. SERE (editors), Lecture Notes in Computer Science, n^{o} 5014, Springer Verlag, May 2008, p. 278–293, http://hal.inria.fr/inria-00259944.

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

Year Publications

Doctoral Dissertations and Habilitation Theses

[15] J. STOECKER. *Un modèle intermédiaire pour la vérification des systèmes asynchrones embarqués temps réel : définition et application du langage ATLANTIF*, Grenoble INP, France, December 2009, Ph. D. Thesis.

Articles in International Peer-Reviewed Journal

[16] H. GARAVEL, G. SALAÜN, W. SERWE. *On the Semantics of Communicating Hardware Processes and their Translation into LOTOS for the Verification of Asynchronous Circuits with CADP*, in "Science of Computer Programming", vol. 74, n^{o} 3, January 2009, p. 100–127, http://hal.inria.fr/inria-00381642.

[17] C. HELMSTETTER, F. MARANINCHI, L. MAILLET-CONTOZ. *Full Simulation Coverage for SystemC Transaction-Level Models of Systems-on-a-Chip*, in "Formal Methods in System Design", vol. 35, n^{o} 2, June 2009, p. 152–189, http://hal.inria.fr/hal-00429058.

[18] F. LANG, G. SALAÜN, R. HÉRILIER, J. KRAMER, J. MAGEE. *Translating FSP into LOTOS and Networks of Automata*, in "Formal Aspects on Computing", 2010, to appear.

[19] P. T. MONTEIRO, E. DUMAS, B. BESSON, R. MATEESCU, M. PAGE, A. T. FREITAS, H. DE JONG. *A Service-Oriented Architecture for Integrating the Modeling and Formal Verification of Genetic Regulatory Networks*, in "BMC Bioinformatics", vol. 10, n^{o} 450, December 2009, http://hal.inria.fr/inria-00440817.

[20] A. WIJS, J. VAN DE POL, E. BORTNIK. *Solving Scheduling Problems by Untimed Model Checking — The Clinical Chemical Analyser Case Study*, in "International Journal on Software Tools for Technology Transfer", vol. 11, n^{o} 5, November 2009, p. 375–392.

International Peer-Reviewed Conference/Proceedings

[21] C. ANDRÉ, M. BELAUNDE, B. BERTHOMIEU, C. BRUNETTE, A. CANALS, H. GARAVEL, S. GRAF, F. LANG, V. MAHÉ, M. NAKHLÉ, R. SCHNECKENBURGER, R. DE SIMONE, J.-P. TALPIN, F. VERNADAT. *Les résultats du projet OpenEmbeDD*, in "Actes de la conférence NEPTUNE'2009 (Paris, France)", A. CANALS, T. MILLAN, J.-C. RAULT (editors), Génie Logiciel, vol. 89, AFCET, June 2009, p. 25–30, http://hal.inria.fr/inria-00381639.

[22] G. CHEHAIBAR, M. ZIDOUNI, R. MATEESCU. *Modeling Multiprocessor Cache Protocol Impact on MPI Performance*, in "Proceedings of the 2009 IEEE International Workshop on Quantitative Evaluation of Large-

Scale Systems and Technologies QuEST'09 (Bradford, UK)", IEEE Computer Society Press, May 2009, http://hal.inria.fr/inria-00381674.

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

[24] S. EDELKAMP, V. SCHUPPAN, D. BOSNACKI, A. WIJS, A. FEHNKER, H. ALJAZZAR. *Survey on Directed Model Checking*, in "Proceedings of the 5th International Workshop on Model Checking and Artificial Intelligence MoChArt'2008 (Patras, Greece)", Lecture Notes in Computer Science, vol. 5348, Springer Verlag, 2009, p. 65–89, http://hal.inria.fr/inria-00406552.

[25] H. GARAVEL, C. HELMSTETTER, O. PONSINI, W. SERWE. *Verification of an Industrial SystemC/TLM Model using LOTOS and CADP*, in "Proceedings of the 7th ACM-IEEE International Conference on Formal Methods and Models for Codesign MEMOCODE'2009 (Cambridge, MA, USA)", IEEE Computer Society Press, June 2009, http://hal.inria.fr/inria-00408283.

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

[27] V. JOLOBOFF, C. HELMSTETTER, H. XIAO. *SimSoC: A full System Simulation Software for Embedded Systems*, in "Proceedings of the International Workshop on Open-Source Software for Scientific Computation OSSC'2009 (Guiyang, China)", IEEE Computer Society Press, September 2009, http://hal.inria.fr/inria-00435247.

[28] F. LANG, R. MATEESCU. *Partial Order Reductions using Compositional Confluence Detection*, in "Proceedings of the 16th International Symposium on Formal Methods FM'2009 (Eindhoven, The Netherlands)", J. BAETEN, A. CAVALCANTI, D. DAMS (editors), Lecture Notes in Computer Science, vol. 5850, Springer Verlag, November 2009, p. 157–172, http://hal.inria.fr/inria-00423583.

[29] R. MATEESCU, A. WIJS. *Hierarchical Adaptive State Space Caching based on Level Sampling*, in "Proceedings of the 15th International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS'2009 (York, UK)", S. KOWALEWSKI, A. PHILIPPOU (editors), Lecture Notes in Computer Science, vol. 5505, Springer Verlag, March 2009, p. 215–229, http://hal.inria.fr/inria-00381682.

[30] J. STOECKER, F. LANG, H. GARAVEL. *Parallel Processes with Real-Time and Data: The ATLANTIF Intermediate Format*, in "Proceedings of the 7th International Conference on Integrated Formal Methods IFM'09 (Düsseldorf, Germany)", M. LEUSCHEL, H. WEHRHEIM (editors), Lecture Notes in Computer Science, vol. 5423, Springer Verlag, February 2009, p. 88–102, http://hal.inria.fr/inria-00391024.

Research Reports

[31] R. MATEESCU, S. MERIOT, S. RAMPACEK. *Extending SPARQL with Temporal Logic*, n^{o} RR-7056, INRIA, October 2009, http://hal.inria.fr/inria-00404761, Research Report.

[32] R. MATEESCU, A. WIJS. *Efficient On-the-Fly Computation of Weak Tau-Confluence*, n^{o} RR-7000, INRIA, July 2009, http://hal.inria.fr/inria-00407381, Research Report.

[33] J. VAN DE POL, R. MATEESCU, H. DE JONG, L. BRIM. *Property-Dependent, On-the-Fly Optimizations of Automata*, FP6-NEST-STREP 043235 project EC-MOAN, July 2009, Project deliverable D.3.3.

Scientific Popularization

[34] H. GARAVEL. *Validating Multiprocessor Architectures*, May 2009, http://www.inria.fr/actualites/inedit/mai09/multiprocesseurs.en.html, INédit 69, INRIA.

Other Publications

[35] X. CLERC, D. CHAMPELOVIER, H. GARAVEL, Y. GUERTE, W. SERWE, G. SMEDING. *Reference Manual of the* LOTOS NT *to* LOTOS *Translator — Version 4K*, 2009, Inria/Vasy, 101 pages.

[36] C. HELMSTETTER. *TLM.OPEN: a SystemC/TLM Front-End for the CADP Verification Toolbox*, October 2009, http://hal.inria.fr/hal-00429070, Workshop on Simulation Based Development of Certified Embedded Systems SBDCES'09 (Awaji Island, Hyogo, Japan).

References in notes

[37] M. ALPUENTE, M. FELIÚ, C. JOUBERT, A. VILLANUEVA. *DATALOG_SOLVE: A Datalog-Based Demand-Driven Program Analyzer*, in "Proceedings of the 8th Spanish Conference on Programming and Computer Languages PROLE'2009 (Gijón, Spain)", J. ALMENDROS-JIMÉNEZ (editor), Electronic Notes in Theoretical Computer Science, vol. 248, Elsevier, August 2009, p. 57–66.

[38] L. APVRILLE. *TTool for DIPLODOCUS: An Environment for Design Space Exploration*, in "Proceedings of the 8th International Conference on New Technologies in Distributed Systems NOTERE'08 (Lyon, France)", ACM Computer Society Press, June 2008, p. 1–4.

[39] M. ATIF. *Formal Analysis of Consensus Protocols in Asynchronous Distributed Systems*, n^{o} 09-16, Technical University of Eindhoven, 34 pages, October 2009, Computer Science Report.

[40] M. ATIF, M. MOUSAVI. *Formal Specification and Analysis of Accelerated Heartbeat Protocols*, n^{o} 09-04, Technical University of Eindhoven, 24 pages, March 2009, Computer Science Report.

[41] T. BARROS, R. AMEUR-BOULIFA, A. CANSADO, L. HENRIO, E. MADELAINE. *Behavioural Models for Distributed Fractal Components*, in "Annals of Telecommunications", vol. 64, n^{o} 1–2, February 2009, p. 25–43.

[42] 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.

[43] 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 2.0*, AESE (pôle de compétitivité mondial Midi-Pyrénées & Aquitaine: Aéronautique, Espace et Systèmes Embarqués) project Topcased, April 2007, Project deliverable F3.2.2 (updated).

[44] 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.

[45] S. BLOM, S. ORZAN. *Distributed State Space Minimization*, in "Springer International Journal on Software Tools for Technology Transfer (STTT)", vol. 7, n^{o} 3, 2005, p. 280–291.

[46] E. BODE, T. PEIKENKAMP, J. RAKOW, S. WISCHMEYER. *Model Based Importance Analysis for Minimal Cut Sets*, in "Proceedings of the 6th International Symposium on Automated Technology for Verification and Analysis ATVA'08 (Seoul, South Korea)", S. D. CHA, J.-Y. CHOI, M. KIM, I. LEE, M. VISWANATHAN (editors), Lecture Notes in Computer Science, vol. 5311, Springer Verlag, October 2008, p. 303–317.

[47] H. BOUDALI, P. CROUZEN, M. STOELINGA. *A Rigorous, Compositional, and Extensible Framework for Dynamic Fault Tree Analysis*, in "IEEE Transactions on Dependable and Secure Computing", vol. 99, n^{o} 1, 2009.

[48] H. BOUDALI, H. SÖZER, M. STOELINGA. *Architectural Availability Analysis of Software Decomposition for Local Recovery*, in "Proceedings of the 3rd IEEE International Conference on Secure Software Integration and Reliability Improvement SSIRI'2009 (Shanghai, China)", IEEE Computer Society Press, July 2009, p. 14–22.

[49] A. CANSADO, E. MADELAINE. *Specification and Verification for Grid Component-Based Applications: From Models to Tools*, in "Proceedings of the 7th International Symposium on Formal Methods for Components and Objects FMCO'2008 (Sophia Antipolis, France)", F. DE BOER, M. BONSANGUE, E. MADELAINE (editors), Lecture Notes in Computer Science, vol. 5751, Springer Verlag, August 2009, p. 180–203.

[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] P. CROUZEN, J. VAN DE POL, A. RENSINK. *Applying Formal Methods to Gossiping Networks with mCRL and Groove*, in "SIGMETRICS Performance Evaluation Review", vol. 36, n^{o} 3, 2008, p. 7–16.

[52] J. CÁMARA, J. A. MARTÍN, G. SALAÜN, J. CUBO, M. OUEDERNI, C. CANAL, E. PIMENTEL. *ITACA: An Integrated Toolbox for the Automatic Composition and Adaptation of Web Services*, in "Proceedings of the 31st International Conference on Software Engineering ICSE'09 (Vancouver, Canada)", IEEE Computer Society Press, May 2009, p. 627–630.

[53] J. CÁMARA, G. SALAÜN, C. CANAL, M. OUEDERNI. *Interactive Specification and Verification of Behavioural Adaptation Contracts*, in "Proceedings of the 9th International Conference on Quality Software QSIC'09 (Jeju, Korea)", IEEE Computer Society Press, August 2009, p. 65–75.

[54] 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.

[55] W. FOKKINK, P. KLINT, B. LISSER, Y. USENKO. *Towards Formal Verification of ToolBus Scripts*, in "Proceedings of the 12th International Conference on Algebraic Methodology and Software Technology

AMAST'2008 (Urbana, IL, USA)", J. MESEGUER, G. ROSU (editors), Lecture Notes in Computer Science, vol. 5140, Springer Verlag, July 2008, p. 160–166.

[56] B. FONTAN, P. DE SAQUI-SANNES, L. APVRILLE. *Synthèse d'observateurs à partir d'exigences temporelles*, in "Proceedings of the 14th Conference on Languages and Models with Objects LMO'08 (Montréal, Canada)", M. BLAY-FORNARINO, Y.-G. GUÉHENEUC, H. SAHRAOUI (editors), March 2008.

[57] 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.

[58] J. F. GROOTE, M. P. A. SELLINK. *Confluence for Process Verification*, in "Theoretical Computer Science", vol. 170, n^o 1–2, 1996, p. 47–81.

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

[60] H. HERMANNS. *Interactive Markov Chains and the Quest for Quantified Quality*, LNCS, vol. 2428, Springer Verlag, 2002.

[61] IEEE. *IEEE Standard SystemC Language Reference Manual*, n^o 1666-2005, Institution of Electrical and Electronic Engineers, December 2005, http://standards.ieee.org/getieee/1666/download/1666-2005.pdf, IEEE Standard.

[62] R. JAIN. *The Art Of Computer Systems Performance Analysis*, Wiley, April 1991.

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

[64] G. KLYNE, J. J. CARROLL. *Resource Description Framework (RDF): Concepts and Abstract Syntax*, W3C, February 2004, W3C Recommendation.

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

[66] X. LI, S. MADNICK, H. ZHU, Y. FAN. *Improving Data Quality for Web Services Composition*, in "Proceedings of the 7th International Workshop on Quality in Databases QDB'2009 (Lyon, France)", August 2009.

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

[68] J. MARKOVSKI, E. DE VINK. *Performance Evaluation of Distributed Systems Based on a Discrete Real- and Stochastic-Time Process Algebra*, in "Fundamenta Informaticae", vol. 95, n^o 1, 2009, p. 157–186.

[69] K. L. MCMILLAN. *The SMV Language*, Cadence Berkeley Labs, March 1999, Technical report.

[70] J. V. MEULEN, C. PECHEUR. *Efficient Symbolic Model Checking for Process Algebras*, in "Proceedings of the 13th International Workshop on Formal Methods for Industrial Critical Systems FMICS'2008 (L'Aquila, Italy)", D. COFER, A. FANTECHI (editors), Lecture Notes in Computer Science, vol. 5596, Springer Verlag, September 2008, p. 69–84.

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

[72] M. MOY, F. MARANINCHI, L. MAILLET-CONTOZ. *LusSy: A Toolbox for the Analysis of Systems-on-a-Chip at the Transactional Level*, in "Proceedings of the 5th International Conference on Application of Concurrency to System Design ACSD'2005 (Saint Malo, France)", IEEE Computer Society, June 2005, p. 26–35.

[73] M. MOY. *Techniques and Tools for the Verification of Systems-on-a-Chip at the Transaction Level*, INPG, Grenoble, France, December 2005, http://www-verimag.imag.fr/~moy/phd/, Ph. D. Thesis.

[74] E. PRUD'HOMMEAUX, A. SEABORNE. *SPARQL Query Language for RDF*, World Wide Web Consortium, January 2008, W3C Recommendation.

[75] R. RIETEMA. *Automatic Verification and Analysis of Test Results of Océ Printers*, University of Twente, The Netherlands, May 2009, Masters thesis.

[76] S. ROOLVINK, A. REMKE, M. STOELINGA. *Dependability and Survivability Evaluation of a Water Distribution Process with Arcade*, in "Proceedings of the 9th International Workshop on Performability Modeling of Computer and Communication Systems PMCCS'2009 (Eger, Hungary)", September 2009.

[77] 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, vol. 4591, Springer Verlag, July 2007, p. 558–578.

[78] G. SALAÜN, T. BULTAN. *Realizability of Choreographies using Process Algebra Encodings*, in "Proceedings of the 7th International Conference on Integrated Formal Methods IFM'09 (Düsseldorf, Germany)", M. LEUSCHEL, H. WEHRHEIM (editors), Lecture Notes in Computer Science, vol. 5423, Springer Verlag, February 2009, p. 167–182.

[79] L. SU, H. BOWMAN, P. BARNARD, B. WYBLE. *Process Algebraic Modelling of Attentional Capture and Human Electrophysiology in Interactive Systems*, in "Formal Aspects of Computing", vol. 21, n^{o} 6, December 2009, p. 513–539.

[80] C. SZABO, Y. M. TEO. *An Approach for Validation of Semantic Composability in Simulation Models*, in "Proceedings of the 23rd ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation PADS'2009 (Lake Placid, NY, USA)", IEEE Computer Society, June 2009, p. 3–10.

[81] H. SÖZER. *Architecting Fault-Tolerant Software Systems*, University of Twente, The Netherlands, January 2009, Ph. D. Thesis.

[82] K. J. TURNER, K. L. L. TAN. *A Rigorous Methodology for Composing Services*, in "Proceedings of the 14th International Workshop on Formal Methods for Industrial Critical Systems FMICS'2009 (Eindhoven, The Netherlands)", M. ALPUENTE, B. COOK, C. JOUBERT (editors), Lecture Notes in Computer Science, vol. 5825, Springer Verlag, November 2009, p. 165–180.

[83] M. WEIGLHOFER, G. FRASERA, F. WOTTAWA. *Using Coverage to Automate and Improve Test Purpose Based Testing*, in "Information and Software Technology", vol. 51, n^{o} 11, November 2009, p. 1601–1617.

[84] P. DE SAQUI-SANNES, L. APVRILLE. *Making Formal Verification Amenable to Real-Time UML Practitioners*, in "Proceedings of the 12th European Workshop on Dependable Computing EWDC'2009 (Toulouse, France)", H. WAESELYNCK (editor), IEEE Computer Society Press, May 2009.

[85] M. DEL MAR GALLARDO, P. MERINO, D. SANÁN. *Model Checking C Programs with Dynamic Memory Allocation*, in "Proceedings of the 32nd Annual IEEE International Computer Software and Applications Conference COMPSAC'2008 (Turku, Finland)", IEEE Computer Society Press, August 2008, p. 219–226.