INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team Pop Art

Programming languages, Operating systems, Parallelism, and Aspects for Real-Time

Grenoble - Rhône-Alpes

THEME COM

Table of contents

1. Team

Project leader
Alain Girault [DR INRIA, HdR]

Project assistants
Helen Pouchot [Secretary (SAR) INRIA]

Inria permanent researchers
Pascal Fradet [CR INRIA, HdR]
Gregor Goessler [CR INRIA]
Bertrand Jeannet [CR INRIA]
Daniel Le Métayer [DR INRIA, HdR]
Eric Rutten [CR INRIA, HdR]

PhD students
Gwenaël Delaval [MENRT grant]
Simplice Djoko Djoko [INRIA grant, AOSD NoE]
Mouaiad Alras [Region Rhône-Alpes grant, ISLE cluster, since 10/2006]
Gérald Vaisman [INRIA grant, DCN contract, since 10/2006]

Post-doctoral fellows
Xavier Briand [MENRT grant, since 11/2007]
Adrien Richard [MENRT grant, until 09/2007]

Interns
Xing Lu [M2R Grenoble, until 09/2007]

Visitor
Rémi Douence [École des Mines de Nantes, until 02/2007]

External Partners
Emil Dumitrescu [INSA Lyon]

2. Overall Objectives

2.1. Introduction

We work on the problem of the safe design of real-time control systems. This area is related to control theory as well as computer science. Application domains are typically safety-critical systems, as in transportation (avionics, railways), production, medical or energy production systems. Both methods and formal models for the construction of correct systems, as well as their implementation in computer assisted design tools, targeted to specialists of the applications, are needed. We contribute to propose solutions all along the design flow, from the specification to the implementation: we develop techniques for the specification and automated generation of safe real-time executives for control systems, as well as static analysis techniques to check additional properties on the generated systems. Our special research themes are:

- implementations of synchronous reactive programs, generated automatically by compilation, particularly from the point of view of distribution (in relation with the LUSTRE [1] and ESTEREL [2] languages) and fault tolerance (in relation with the SYNDEX [3] environment);

[1] http://www-verimag.imag.fr/SYNCHRONE
[2] http://www.inria.fr/recherche/equipes/aoste.en.html
[3] http://www-rocq.inria.fr/syndex

- high-level design and programming methods, with support for automated code generation, including: the automated generation of correct controllers using discrete control synthesis (in relation with the Mode Automata [4] and SIGNAL [5] languages, and with the SIGALI synthesis tool); compositionality for the verification, and construction of correct systems; reactive programming, aspect-oriented programming.
- static analysis and abstract interpretation techniques, which are applied both to low-level synchronous models/programs and to more general imperative programs; this includes the verification of general safety properties and the absence of runtime errors.

Our applications are in embedded systems, typically in the robotics, automotive, and telecommunications domains with a special emphasis on dependability issues (*e.g.*, fault tolerance, availability). International and industrial relations feature:

- two IST European networks of excellence:
  - ARTIST II [6], about embedded real-time systems,
  - AOSD-Europe[7], about formal methods for Aspect-Oriented Programming,
- three ACIs ("Actions Concertées Incitatives"): ALIDECS (on large-scale critical embedded systems), DISPO (on security policies for software components), and APRON (numerical program analysis);
- the OPENTLM project of the MINALOGIC Pole of Competitiveness, dedicated to the design flow for next generation SoC and SystemC,
- industrial collaborations with DCN and POLYSPACE.

2.2. Highlights of the year

2.2.1. The APRON library: three years of effort

Keywords: *Safety-critical systems*, *abstract interpretation of numerical variables*, *static analysis*.

POP ART has been involved in the ACI "Sécurité et informatique" APRON (Analyse de PROgrammes Numériques) (see 8.2.3), which focused on the theory of numerical abstract domains, and their application to the static analysis of the numerical variables of a program.

The main practical goal of this three-years project was to mature the field by designing and implementing a common software platform suited for a broad range of static analysis applications, after having clarified and unified the needs of the five actors of the project. This work resulted in the APRON library, which is dedicated to the static analysis of the numerical variables of a program by abstract interpretation. Its goal is threefold:

1. providing ready-to-use numerical abstractions for analysis implementers,
2. encouraging the research in numerical abstract domains by providing a platform for integration and comparison,
3. and providing a teaching and demonstration tool to disseminate knowledge on abstract interpretation.

APRON is not tied to a particular numerical abstraction. Several abstract domain implementations providing various precision versus cost trade-off are currently implemented. A specific low-level C API was designed to minimize the effort when incorporating a new abstract domain.

From the point of view of the analysis designer, APRON exposes a higher-level, richer, and language-agnostic API. Bindings for C, C++, and OCaml are currently provided. An important recent inclusion is the treatment of non-linear and floating-point expressions in assignments and tests.

[4] http://www-verimag.imag.fr/PEOPLE/Florence.Maraninchi/MATOU
[5] http://www.irisa.fr/espresso
[6] http://www.artist-embedded.org
[7] http://www.aosd-europe.net/

The APRON library is freely available on the web[8] and is released under the LGPL license. More details can be found in section 5.4.

3. Scientific Foundations

3.1. Embedded systems and their safe design

Keywords: *Embedded systems*, *control*, *distribution*, *real-time*, *safety-criticality*.

3.1.1. The safe design of embedded real-time control systems.

The context of our work is the area of embedded real-time control systems, at the intersection between control theory and computer science. Our contribution consists of methods and tools for their safe design. The systems we consider are intrinsically safety-critical because of the interaction between the embedded, computerized controller, and a physical process having its own dynamics. What is important is to analyze and design the safe behavior of the whole system, which introduces an inherent complexity. This is even more crucial in the case of systems whose malfunction can have catastrophic consequences, for example in transport systems (avionics, trains), production, medical, or energy production systems.

Therefore, there is a need for methods and tools for the design of safe systems. The definition of adequate mathematical models of the behavior of the systems allows the definition of formal calculi. They in turn form a basis for the construction of algorithms for the analysis, but also for the transformation of specifications towards an implementation. They can then be implemented in software environments made available to the users. A necessary complement is the setting-up of software engineering, programming, modeling, and validation methodologies. The motivation of these problems is at the origin of significant research activity, internationally and in particular, in the European IST network of excellence ARTIST II (Advanced Real-Time Systems)[9].

3.1.2. Models, methods and techniques.

The state of the art upon which we base our contributions, is twofold.

From the point of view of discrete control, there is a set of theoretical results and tools, in particular in the synchronous approach, often founded on labeled transition systems finite or infinite [34], [43]. During the past years, methodologies for the formal verification [72], [45], control synthesis [74] and compilation, and extensions to timed and hybrid systems [68], [35] have been developed. Asynchronous models consider the interleaving of events or messages, and are often applied in the field of telecommunications, in particular for the study of protocols. A well-known formalism for reactive systems is STATECHARTS [63], which can be encoded in a synchronous model [37].

From the point of view of verification, we use the methods and tools of symbolic model-checking and of abstract interpretation. From symbolic model-checking, we reuse BDD techniques [39] for manipulating Boolean functions and sets, and their MTBDD extension for more general functions. Abstract Interpretation [47] is used to formalize complex static analysis, in particular when one wants to analyze the possible values of variables and pointers of a program. Abstract Interpretation is a theory of approximate solving of fix-point equations applied to program analysis. Most program analysis problems, among others reachability analysis, come down to solving a fix-point equation on the state space of the program. The exact computation of such an equation is generally not possible for undecidability (or complexity) reasons. The fundamental principles of Abstract Interpretation are: (i) to substitute to the state-space of the program a simpler domain and to transpose the equation accordingly (static approximation); and (ii) to use extrapolation (widening) to force the convergence of the iterative computation of the fix-point in a finite number of steps (dynamic approximation). Examples of static analysis based on abstract interpretation are the Linear Relation Analysis [48] and Shape Analysis [44].

[8] http://apron.cri.ensmp.fr/library
[9] http://www.systemes-critiques.org/ARTIST

The synchronous approach [10] [61], [62] to reactive systems design gave birth to complete programming environments, with languages like ARGOS, LUSTRE [11], ESTEREL [12], SIGNAL/ POLYCHRONY [13], SYNDEX [14], LUCID SYNCHRONE [15] or Mode Automata [16]. This approach is characterized by the fact that it considers periodically sampled systems whose global steps can, by synchronous composition, encompass a set of events (known as simultaneous) on the resulting transition. Generally speaking, formal methods are often used for analysis and verification; they are much less often integrated in the compilation or generation of executives (in the sense of executables of tasks combined with the host real-time operating system). They are notoriously difficult to use by end-users, who are usually specialists in the application domain, not in formal techniques. This is why encapsulating formal techniques in an automated framework can dramatically improve their diffusion, acceptance, and hence impact. Our work is precisely oriented towards this direction.

3.2. Issues in design automation for complex systems

Keywords: *compilation*, *design automation*, *formal methods*, *real-time executives*, *scheduling*, *synthesis*, *verification*.

3.2.1. Hard problems

The design of safe real-time control systems is difficult due to various issues, among them their complexity in terms of the number of interacting components, their parallelism, the difference of the considered time scales (continuous or discrete), and the distance between the various theoretical concepts and results that allow the study of different aspects of their behaviors, and the design of controllers. The European IST network of excellence ARTIST II identifies three principal objectives: hard real-time for critical applications (which concerns the synchronous approach), component-based design, and adaptive real-time systems for quality of service management.

A currently very active research direction focuses on the models and techniques that allow the automatic use of formal methods. In the field of verification, this concerns in particular the technique of model checking; the verification takes place after the design phase, and requires, in case of problematic diagnostics, expensive backtracks on the specification. We want to provide a more constructive use of formal models, using them to derive correct executives by formal computation and synthesis, integrated in a compilation process. We therefore use models throughout the design flow from specification to implementation, in particular by automatic generation of embeddable executives.

3.2.2. Applicative needs

They initially come from the fields of safety-critical systems (avionics, energy) and complex systems (telecommunication), embedded in an environment with which they strongly interact (comprising aspects of computer science and control theory). Fields with less strong criticality, or which support variable degrees of quality of service, such as in the multi-media domain, can also take advantage of methodologies that improve the quality and reliability of software, and reduce the costs of test and correction in the design.

Industrial acceptance, the dissemination, and the deployment of the formal techniques inevitably depend on the usability of such techniques by specialists in the application domain — and not in formal techniques themselves —, and also on the integration in the whole design process, which concerns very different problems and techniques. The application domains are rather rare where the actors are ready to employ specialists in formal methods or advanced control theory. Even then, the methods of systematic application of these theoretical results are not ripe. In fields like industrial control, where the use of PLC (Programmable Logic Controller [40]) is dominant, this question can be decisive.

[10] http://www.synalp.org
[11] http://www-verimag.imag.fr/SYNCHRONE
[12] http://www.inria.fr/recherche/equipes/aoste.en.html
[13] http://www.irisa.fr/espresso/Polychrony
[14] http://www-rocq.inria.fr/syndex
[15] http://www.lri.fr/~pouzet/lucid-synchrone/
[16] http://www-verimag.imag.fr/PEOPLE/Florence.Maraninchi/MATOU

Essential elements in this direction are the proposal of realistic formal models, validated by experiments, of the usual entities in control theory, and functionalities (*i.e.*, algorithms) that correspond indeed to services useful for the designer. Take for example the compilation and optimization taking into account the platforms of execution, possible failures, or the interactions between the defined automatic control and its implementation. A notable example for the existence of an industrial need is the activity of the ATHYS company (now belonging to DASSAULT SYSTEME) concerning the development of a specialized programming environment, CELLCONTROL, which integrates synchronous tools for compilation and verification, tailored to the application domain. In these areas, there are functionalities that commercial tools do not have yet, and to which our results contribute.

3.2.3. Our approach

We are proposing effective trade-offs between, on the one hand, expressiveness and formal power, and on the other, usability and automation. We focus on the area of specification and construction of correct real-time executives for discrete and continuous control, while keeping an interest in tackling major open problems, relating to the deployment of formal techniques in computer science, especially at the border with control theory. Regarding the applications, we propose new automated functionalities, to be provided to the users in integrated design and programming environments.

3.3. Main Research Directions

Keywords: *aspect-oriented programming*, *compositionality*, *controller generation*, *dedicated languages*, *distribution*, *fault tolerance*.

3.3.1. Principles

We intend to exploit our knowledge of formal techniques and their use, and of control theory, according to aspects of the definition of fundamental tools, and applications.

The integration of formal methods in an automated process of generation/compilation is founded on the formal modeling of the considered mechanisms. This modeling is the base for the automation, which operates on models well-suited for their efficient exploitation, by analysis and synthesis techniques that are difficult to use by end-users.

The creation of easily usable models aims at giving the user the role rather of a pilot than of a mechanics *i.e.*, to offer her/him pre-defined functionalities which respond to concrete demands, for example in the generation of fault tolerant or distributed executives, by the intermediary use of dedicated environments and languages.

The proposal of validated models with respect to their faithful representation of the application domain is done through case studies in collaboration with our partners, where the typical multidisciplinarity of questions across control theory and computer science is exploited.

The overall consistency of our approach comes from the fact that the main research directions address, under different aspects, the specification and generation of safe real-time control executives based on *formal models*.

We explore this field by linking, on the one hand, the techniques we use, with on the other, the functionalities we want to offer. We are interested in questions related to:

- dedicated languages and models for automatic control that are the interface between the techniques we develop and the end-users on the one hand, and the designers of formal models on the other;
- compositional modeling and analysis that aim at deriving crucial system properties from component properties, without the need to actually build and check the global system;
- static analysis and abstract interpretation methods for checking functional properties on models and generated programs;
- Aspect-Oriented Programming (AOP) that allows to express safety concerns separately from the functional part and to enforce them on programs.

3.3.2. Implementations of synchronous programs

This issue can be tackled differently depending on the execution platform. Based on a formal model of the program to be implemented, our approach is to obtain by compilation (*i.e.*, automatically):

- the distribution on a multiprocessor architecture, with code partitioning according to directives, and insertion of the necessary communication actions to ensure the coherence of control; the distribution must be correct with respect to the original specification, and must be optimized;
- fault tolerance by replication of computations on a multiprocessor architecture, and scheduling of computations according to the faults to be tolerated; such a scheduling must be optimized *w.r.t.* its length and reliability.

3.3.3. Automatic generation of correct controllers

We use techniques of discrete controller synthesis, especially the tools SIGALI [70] and Mode Automata [69] within an automated framework, for:

- multi-mode multi-tasking systems where the management of interactions (exclusions, optimization of cost or quality criteria, ...) is obtained by synthesis;
- a locally imperative, globally declarative language whose compilation comprises a phase of discrete controller synthesis;
- fault-tolerance management, by reconfiguration following objectives of consistent execution, functionality fulfillment, boundedness and optimality of response time;
- and, more generally, a model-based approach to adaptive systems, with applications in embedded middleware for autonomic systems, and reconfigurable architectures.

4. Application Domains

4.1. Industrial applications.

Our applications are in embedded systems, typically: robotics, automotive, telecommunications, systems on chip (SoC). In some areas, safety is critical, and motivates the investment in formal methods and techniques for design. But even in less critical contexts, like telecommunications and multimedia, these techniques can be beneficial in improving the efficiency and quality of designs, as well as the design, production and test costs themselves.

Industrial acceptance of formal techniques, as well as their deployment, goes necessarily through their usability by specialists of the application domain, rather than of the formal techniques themselves. Hence our orientation towards the proposal of domain-specific (but generic) realistic models, validated through experience (*e.g.*, control tasks systems), based on formal techniques with a high degree of automation (*e.g.*, synchronous models), and tailored for concrete functionalities (*e.g.*, code generation).

4.2. Industrial design tools

The commercially available design tools (such as UML with real-time extensions, MATLAB/ SIMULINK/ dSPACE [17]) and execution platforms (OS such as VXWORKS, QNX, real-time versions of LINUX ...) propose a collection of functionalities without accompanying it by design or verification methods. Some of them, founded on models of reactive systems, come close to tools with a formal basis, such as for example STATEMATE by iLOGIX.

[17] http://www.dspaceinc.com

Regarding the synchronous approach, commercial tools are available: SCADE (based on LUSTRE), ESTEREL [18], SILDEX [19] (based on SIGNAL), specialized environments like CELLCONTROL for industrial automatism (by the INRIA spin-off ATHYS). One can note that behind the variety of actors, there is a real consistency of the synchronous technology, which makes sure that the results of our work related to the synchronous approach are not restricted to some language due to compatibility issues.

4.3. Current industrial cooperations.

Regarding applications and case studies with industrial end-users of our techniques, we cooperate with STMicroelectronics on compositional analysis and abstract interpretation for the TLM-based System-on-Chip design flow, and with DCN on the multi-criteria real-time scheduling issues for action planning of their defense systems.

5. Software

5.1. NBac

Participant: B. Jeannet [contact person].

NBAC (Numerical and Boolean Automaton Checker)[20] is a verification/slicing tool for reactive systems containing combination of Boolean and numerical variables, and continuously interacting with an external environment. NBac can also handle the same class of hybrid systems as the HyTech tool. It aims at handling efficiently systems combining a non-trivial numerical behaviour with a complex logical (Boolean) behaviour.

NBAC is connected to 2 input languages: the synchronous dataflow language LUSTRE, and a symbolic automaton-based language, AUTOC/AUTO, where a system is defined by a set of symbolic hybrid automata communicating via valued channels. It can perform reachability analysis, co-reachability analysis, and combination of the above analyses. The result of an analysis is either a verdict to a verification problem, or a set of states together with a necessary condition to stay in this set during an execution. NBAC is founded on the theory of abstract interpretation: sets of states are approximated by abstract values belonging to an abstract domain, on which fix-point computations are performed.

It has been used for verification and debugging of LUSTRE programs [65] [52]. It is connected to the LUSTRE toolset [21] It has also been used for controller synthesis of infinite-state systems The fact that the analyses are approximated results simply in the obtention of a possibly non-optimal controller. In the context of conformance testing of reactive systems, it is used by the test generator STG [46] [66] for selecting test cases.

5.2. Prometheus

Participant: G. Goessler [contact person].

The BIP component model (Behavior, Interaction model, Priority) [57] [58] has been designed to support the construction of heterogeneous reactive systems involving different models of computation, communication, and execution, on different levels of abstraction. By separating the notions of behavior, interaction model, and execution model, it enables both heterogeneous modeling, and separation of concerns.

The verification and design tool Prometheus implements the BIP component framework. Prometheus is regularly updated to implement new developments in the framework and the analysis algorithms. It has allowed us to carry out several complex case studies from the system-on-chip and bioinformatics domains.

[18] http://www.esterel-technologies.com
[19] http://www.tni-valiosys.com
[20] http://pop-art.inrialpes.fr/people/bjeannet/nbac/
[21] http://www-verimag.imag.fr/SYNCHRONE/index.php?page=tools

5.3. Implementations of synchronous programs

Participants: A. Girault [contact person], H. Kalla.

5.3.1. Code distribution

OCREP distributes automatically synchronous programs according to specifications given by the user. Concretely, starting from a centralized source synchronous program obtained either with the LUSTRE or the ESTEREL compiler, from a number of desired computing locations, and an indication of where each input and output of the source program must be computed, OCREP produces several programs, one for each location, each one computing only its assigned variables and outputs, and communicating harmoniously. Their combined behavior is equivalent to the behavior of the centralized source program and that there is no deadlock.

Currently our software OCREP is distributed in the form of executable on the web[22]. It consists in 15000 lines of C++ code. In 2002, a contract for industrial transfer was drawn up with France Télécom R&D in order to integrate OCREP into their compiler SAXO-RT for ESTEREL programs.

5.3.2. Fault tolerance

We have been cooperating for several years with the INRIA team AOSTE (INRIA Sophia-Antipolis and Rocquencourt) on the subject of fault tolerance. In particular, we have implemented several new heuristics for fault tolerance and reliability within their software SYNDEX [23]. This has taken place within the framework of the European project EAST-EEA in which we participated together with AOSTE. In this context, we have developed several new scheduling heuristics that produce static multiprocessor schedules tolerant to a specified number of processor and communication link failures [53]. The basic principles upon which we rely to make the schedules fault tolerant is, on the one hand, the active replication of the operations [8], and on the other hand, the active replication of communications for point-to-point communication links, or their passive replication coupled with data fragmentation for multi-point communication media (*i.e.*, buses) [54].

5.4. APRON libray

The APRON library[24] is dedicated to the static analysis of the numerical variables of a program by Abstract Interpretation [47]. Its goal is threefold: provide ready-to-use numerical abstractions for analysis implementers, encourage the research in numerical abstract domains by providing a platform for integration and comparison, and provide a teaching and demonstration tool to disseminate knowledge on abstract interpretation.

5.4.1. Motivation and Principles.

Many abstract domains have been designed and implemented for analysing the possible values of numerical variables during the execution of a program, *cf.* Fig. 1. However their API diverge largely (datatypes, signatures, ...), which does not facilitate their diffusion and experimental comparison w.r.t. efficiency and precision aspects.

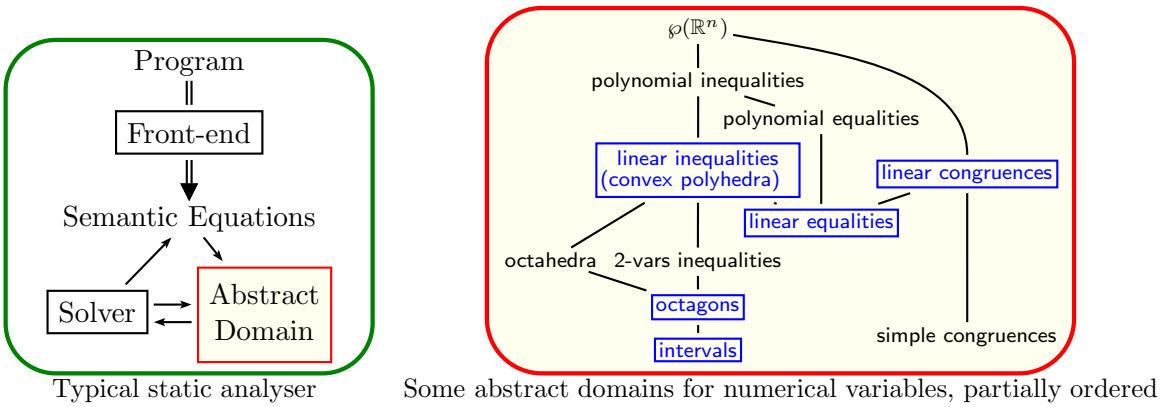
The APRON library aims to provide:

- A uniform API for existing numerical abstract domains;
- A higher-level interface to the client tools, by factorizing functionalities that are largely independent of abstract domains.

From an abstract domain implementor point of view, the benefits of the APRON library are:

- The ability to focus on core, low-level functionalities;
- The help of generic services adding higher-level services for free.

[22] http://pop-art.inrialpes.fr/people/girault/Ocrep/
[23] http://www-rocq.inria.fr/syndex
[24] http://apron.cri.ensmp.fr/library/



Typical static analyser

Some abstract domains for numerical variables, partially ordered w.r.t. their expressiveness

Figure 1. Typical static analyser and examples of abstract domains

For the client static analysis community, the benefits are an unified, higher-level interface, allows experimenting, comparing and combining abstract domains.

5.4.2. Implementation.

Fig. 2 depicts the organisation of the APRON library. The existing underlying libraries connect to the developper interface, using domain-independent datatypes, and exploiting common services. Independent libraries like PPL [36] can be connected using a wrapper. Client tools connect to the higher-level user interface, where variables (or addresses) and environments replace geometrical notions like dimensions and space dimensionality.

The APRON library is written in C ANSI, with an object-oriented and thread-safe design. Both multi-precision and floating-point numbers are supported. A wrapper for the OCaml language is available, and a C++ wrapper is on the way. It is distributed since june 2006 under the LGPL license and available at http://apron.cri.ensmp.fr/.

Its developpement has still progressed much since. There are already many external users (ProVal/Démons, LRI Orsay, France — Analysis of Computer Systems Group, New-York University, USA — Sierum software analysis platform, Kansas State University, USA — NEC Labs, Princeton, USA — EADS CCR, Paris, France — IRIT, Toulouse, France)

5.5. Prototypes

5.5.1. Automatic Controller Generation

Participants: G. Delaval, E. Dumitrescu, A. Girault, E. Rutten [contact person].

We have developed a software tool chain to allow the specification of models, the controller synthesis, and the execution or simulation of the results. It is based on existing synchronous tools, and thus consists primarily in the use and integration of SIGALI [25] and of Mode Automata [26].

Useful component templates and relevant properties can be materialized, on one hand by libraries of task models, and, on the other hand, by properties and synthesis objectives. A prototype compiler has been developed to demonstrate a domain-specific language, named NEMO, for multi-task controllers (see Section 6.3).

[25] http://www.irisa.fr/vertecs/Logiciels/sigali.html
[26] http://www-verimag.imag.fr

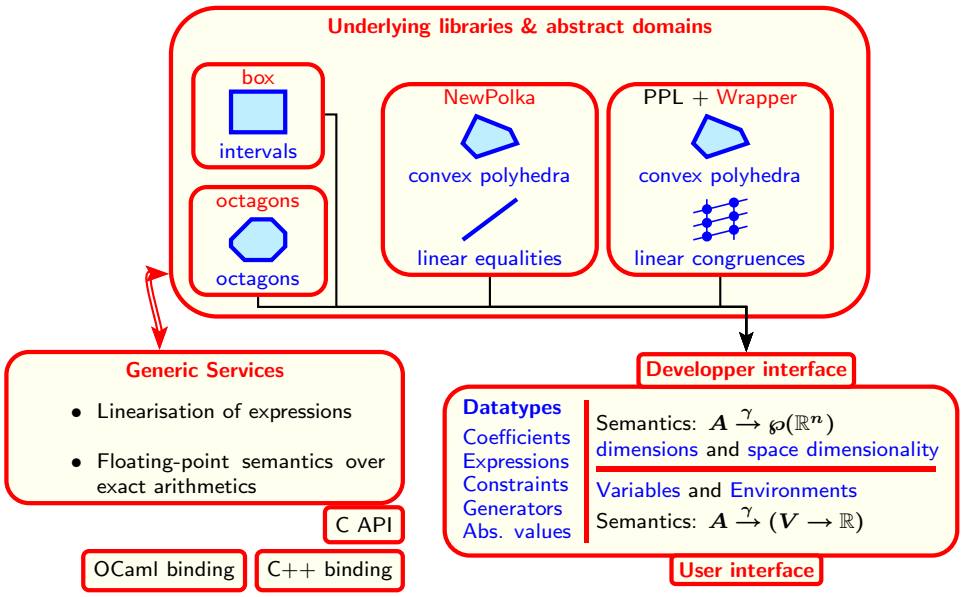


Figure 2. Organisation of the APRON library

5.5.2. *Rapture*

Keywords: *Markov Decision Processes*, *Probabilistic verification*.

Participant: B. Jeannet [contact].

RAPTURE [64] [49] is a verification tool that was developed jointly by BRICS (Denmark) and INRIA in years 2000–2002. The tool is designed to verify reachability properties on Markov Decision Processes (MDP), also known as Probabilistic Transition Systems. This model can be viewed both as an extension to classical (finite-state) transition systems extended with probability distributions on successor states, or as an extension of Markov Chains with non-determinism. We have developed a simple automata language that allows to describe a set of processes communicating over a set of channels *à la* CSP. Processes can also manipulate local and global variables of finite type. Probabilistic reachability properties are specified by defining two sets of initial and final states together with a probability bound. The originality of the tool is to provide two reduction techniques that limit the state space explosion problem: automatic abstraction and refinement algorithms, and the so-called essential states reduction.

5.5.3. *Libraries for Abstract Interpretation*

Participant: B. Jeannet [contact person].

We also develop and maintain smaller libraries of general use for people working in the static analysis and abstract interpretation community.

Fixpoint [27]: a generic fix-point engine written in OCAML. It allows to solve systems of fix-point equations on a lattice, using a parameterized strategy for the iteration order and the application of widening. It also implements very recent techniques [59].

[27] http://bjeannet.gforge.inria.fr/fixpoint

Interproc [28]: a simple interprocedural static analyzer that infers properties on the numerical variables of programs in a toy language. It is aimed at demonstrating the use of the previous library and the above-described APRON library, and more generally at disseminating the knowledge in abstract interpretation. It is also deployed through a web-interface: [29].

6. New Results

6.1. Dependable distributed real-time embedded systems

Participants: P.-F. Dutot, A. Girault [contact person], H. Kalla, G. Vaisman.

6.1.1. Revisiting the bicriteria (length,reliability) multiprocessor static scheduling problem

Our starting point is a dependency task graph and an heterogeneous distributed memory target architecture. We have revisited the well studied problem of bicriteria (length,reliability) multiprocessor static scheduling of this task graph onto this architecture. Our first criteria remains the static schedule's length: this is crucial to assess the system's real-time property. For our second criteria, we have considered the **global system failure rate** (GSFR), seen as if the whole system were a single task scheduled onto a single processor, instead of the usual reliability. Th reason for this choice is that the GSFR does not depend on the schedule length like the reliability does, due to its computation in the classical reliability model of Shatz and Wang [75]. Under this widely accepted reliability model, the probability that a processor be operational during a duration d is $e^{-\lambda d}$, where λ is the failure rate per time unit of this processor (in other words, this is a constant parameter Poisson law). We have shown that, unfortunately, using as the two criteria the length and the reliability yields counter-intuitive results: for instance, choosing a processor such that the duration d of a given operation is smaller (which is good for the length criterion) induces a higher reliability (which is also good for the reliability criterion); this is because the $d \mapsto e^{-\lambda d}$ function is decreasing. This is counter-intuitive because it means that replication is bad for reliability! It follows that it is difficult to design a satisfactory bicriteria scheduling heuristic. In particular, this has three drawbacks: first, the length criterion overpowers the reliability criterion; second, it is very tricky to control precisely the replication factor of the operations onto the processors, from the beginning to the end of the schedule (in particular, it can cause a "funnel" effect); and third, the reliability is not a monotonous function of the schedule.

Instead, by using the GSFR jointly with the schedule length, we have shown that we control better the replication factor of each individual task of the dependency task graph given as a specification, with respect to the desired failure rate. Intuitively, this is because the GSFR is the reliability "per time unit", hence independent of the length. In particular, our new scheduling algorithm does not suffer from the drawbacks mentioned above.

To solve this (length,GSFR) bicriteria optimization problem, we have taken the failure rate as a constraint, and we have minimized the schedule length. We are thus able to produce, for a given application task graph and multiprocessor architecture, a Pareto curve of non-dominated solutions, among which the user can choose the compromise that fits his requirements best [30].

6.1.2. Static multiprocessor scheduling of tasks with resource constraints

We address here the problem of scheduling tasks that have strong constraints. The studied constraints are physical resources (specific processor, memory), waiting until all the predecessors tasks have terminated (time constraint), and real time. Also, the computation of a schedule must be accomplished in a short delay (one second) and the solution found must be as close as possible to the optimum. We try to optimize several criteria at the same time, even if the optimization of a criterion can have a negative effect on other criteria. We search, therefore, a good compromise between those criteria. The criteria that we use are the ending date of the last task (makespan), the minimization of the used resources, and the computation time of the scheduling. The algorithm chosen to calculate the scheduling is a branch and bound, because it can provide a precise solution as well as an approached solution if it is stopped before its end, while guaranteeing the quality of the obtained solution in comparison with the optimum. A prototype has been already implemented and tested.

[28] http://bjeannet.gforge.inria.fr/interproc
[29] http://pop-art.inrialpes.fr/interproc/interprocweb.cgi

We are currently performing experimentations. On the one hand we are trying different initialization algorithms of the branch and bound, and on the other hand we are evaluating several multi-criterion evaluation functions. Since we have a multiprocessor machine to compute the schedules, several possibilities exist. We can either run *one* parallel branch and bound program, or we can run *several* sequential branch and bound programs in parallel. In the former case, we propose to use BOBPP [30] based on KAAPI [31] in order to build a parallel branch and bound program. This will allow us to compare the benefits of the branch and bound algorithm parallelized on several processors. In the latter case, the programs must "share" the better current solution, so that each program cuts faster the less promising branches.

6.2. Automatic distribution of synchronous programs

Participants: M. Alras, G. Delaval, A. Girault [contact person].

6.2.1. Modular distribution of higher-order dataflow synchronous programs

Synchronous programming languages describe functionally centralized systems, where every value, input, output, or function are always directly available for every operation. However, most embedded systems are nowadays composed of several computing resources. The aim of this work is to provide a language-oriented solution to describe functionally distributed reactive systems. This is the topic of the PhD of Gwenaël Delaval, co-advised by Alain Girault and Marc Pouzet (University of Orsay, LRI).

In order to address this problem, we have extended a synchronous dataflow language with primitives for program distribution. These primitives allow the programmer, on one hand to describe the architecture of the system in terms of symbolic locations representing physical locations and links between them, and on the other hand to express where streams and expressions are located in this architecture.

First, a distributed semantics has been proposed in order to formalize the distributed execution of a program. Then, a type and effects system [76], where types of values are their localizations, has been proposed in order to infer the localization of non-annotated values by means of type inference and to ensure, at compilation time, the consistency of the distribution. Finally, a projection operation allows us to obtain automatically, from a centralized typed program, the local program to be executed by each computing resource. The semantical equivalence of the centralized program and its distributed version through this projection operation has been proven.

This type system, as well as the projection operation, has been implemented within the Lucid Synchrone [42] compiler. This higher-order synchronous language allows the expression of stream of stream functions. The distribution method proposed is performed in a modular way, and thus fits with the compilation of such higher-order features. The aim is, by combining this distribution method together with higher-order features of this languages, to allow the expression of dynamic reconfiguration of a hardware resource by another by sending code through communication channels: such channels being then streams of stream functions.

6.2.2. Model-based development of fault-tolerant embedded systems, code generation for distributed heterogeneous platforms

In the domain of safety-critical embedded systems, tools like Matlab/Simulink (trademarks of The Mathworks Inc.) are available for the automatic generation of application code. However, system aspects like process management, communication, or fault-tolerance mechanisms are not covered by these tools, even though such aspects constitute an essential part of the whole code. We wish to extend the model-based design approach of Matlab/Simulink in order to generate automatically executable fault-tolerant code corresponding to a specific platform, possibly distributed and heterogeneous. The main idea of our work involves the translation of Simulink/Stateflow into the synchronous programming language Lustre, allowing its associated compilers, model-checkers, and abstract interpretation tools to be applied to Simulink/Stateflow designs.

[30] BOBPP: http://bobpp.prism.uvsq.fr
[31] KAAPI: http://kaapi.gforge.inria.fr

A translating tool has been already developed in previous study [41], [78]. Our objective is to go ahead with this development and to improve it from the point of view of fault-tolerance and automatic code distribution algorithms. This will be achieved by adding new constructs to the synchronous language Lustre and by adding new features to the embedded code generation tool itself. This is the topic of the PhD of Mouaiad Alras, co-advised by Alain Girault and Pascal Raymond (CNRS, Verimag).

6.3. Automatic generation of correct controllers

Participants: G. Delaval, E. Dumitrescu, A. Girault, E. Rutten [contact person].

We address the difficulty of safely designing complex system controllers by proposing a method applying formal design techniques to the domain of embedded control systems. Such techniques are considered difficult to use, amongst other things because of the required theoretical competence. A general notion of *hidden formal methods* advocates for fully automated techniques, integrated into a design process and tool. The formal technique we aim to encapsulate into a tool chain is *discrete controller synthesis* [73], and more particularly its adaptation to the synchronous approach [70].

6.3.1. Domain-specific language for application of discrete controller synthesis

We have proposed a simple programming language, called NEMO [13], specific to the domain of multi-task real-time control systems, such as in robotics, automotive or avionics systems. The notion of task is related to the one used in the ORCCAD tool [38]. It can be used to specify a set of resources with usage constraints, a set of tasks that consume them according to various modes, and applications sequencing the tasks. We obtain automatically an application-specific task handler that correctly manages the constraints (if any), through a compilation-like process including a phase of discrete controller synthesis. We use synchronous languages, modeling techniques and tools, particularly the Mode Automata language [69] and the SIGALI synthesis tool [70].

6.3.2. Fault tolerant systems

In order to automatically obtain fault tolerant real-time systems, we investigate a new solution based on the application of discrete controller synthesis (DCS). The real-time systems we consider consist of a set of tasks and a set of distributed, heterogeneous processors. The latter are fail-silent, and an environment model can detail actual fault patterns. We apply DCS with objectives w.r.t. consistent execution, functionality fulfillment, and some optimizations. We construct a manager that ensures fault tolerance by migrating the tasks automatically, upon occurrence of a failure, according to the policy given by the objectives.

We have new results concerning optimal synthesis along paths, and its application to the control of sequences of reconfigurations. Tasks that are interrupted by a fault can be restarted at their last checkpoint, and the control of the configuration restarts the tasks by placing them on processors chosen w.r.t. an objective on the shortest total execution time of the application. We therefore combine, on the one hand, guarantees on the safety of the execution by tolerating faults, and on the other hand, guarantees on the worst case execution time of the resulting dynamically reconfiguring fault tolerant system [29], [19], [20].

This work is conducted in collaboration with H. Marchand (VERTECS team from INRIA Rennes) and E. Dumitrescu (INSA Lyon).

6.3.3. Model-based control of adaptative systems

Embedded systems have to be more and more *adaptive*: they must perform reconfigurations in reaction to changes in their environment, related to resources or dependability. The management of this dynamical adaptivity is approached *e.g.*, in autonomic systems, at middleware level, by sensing the state of a system, deciding upon reconfiguration actions, and performing them. It can be considered as a control loop, on continuous or discrete criteria.

We consider that our previous work gave contributions to different separate issues related to this topic; on the basis of this experience, we are beginning to work on a generalisation of this previous work towards a model-based approach to adaptive systems, with applications in embedded middleware for autonomic systems, and reconfigurable architectures. We see it as an approach to combine adaptivity and predictability, and a method for the safe design of safe execution systems, relying on a technique for the static guarantee of dynamic reconfigurations.

This work is conducted in contact with the SARDES team of INRIA in Grenoble.

6.4. Static Analysis and Abstract Interpretation

6.4.1. Implementation of the APRON library for numerical abstract domains

Participant: B. Jeannet [contact person].

This new result corresponds to the software described in section 5.4, in the context of the ACI-SI APRON (see 8.2.3). Since november 2006, the library has been improved with (many) bug corrections and the addition of several features, in collaboration with A. Miné from ENS Paris:

- Addition of generic functionalities, in particular support for the reduced product of abstract domains;
- Support of non-linear expressions and constraints:
  - Manipulation of arbitrary expressions, with integer, floating-point and real operators, and optional specification of rounding mode;
  - Linearisation of such expressions in interval linear expressions, following [71]
- Support of new domains:
  - Linear equalities, implemented on top of the NewPolka convex polyhedra library;
  - Linear congruences, implemented by the PPL;
  - Reduced product of convex polyhedra and linear congruences.
- New language bindings: C++, and soon JAVA;
- Interprocedural analyzer using the APRON library and demonstrating its features (see section 5.4);

The core APRON library represents now 24000 LOC in C (compared to 10000 last year), to which one should add the code for the OCAML and C++ bindings.

We have several external users, as mentioned in section 5.4, and two new domains should be added by external teams (CEA-LIST, Saclay, France, and Theoretical Science Group, University of Kent, UK).

We have presented a poster at the Static Analysis Symposium (SAS'07) that took place in Lyngby, Denmark, august 2007. This poster has also been displayed at the "Grand Colloque STIC 2007" event, 5–7 november 2007, together with a demo, and we also gave a talk.

6.4.2. Verification of Communication Protocols Using Abstract Interpretation of FIFO queues

Participants: T. Le Gall, B. Jeannet [contact person].

The verification of communication protocols or distributed systems that can be modeled by set of sequential machines communicating via unbounded FIFO channels is the topic of the PhD of Tristan Le Gall. The main challenge of its PhD is the verification of such systems in the case where

- the communicating machines are themselves infinite-state processes;
- the values sent to FIFO channels belong to unbounded datatypes.

The approach we follow is based on the theory of Abstract Interpretation. The applications of such verification techniques are the analysis of communicating protocols, which may contain subtle bugs, the automatic synthesis of controllers for distributed systems in order to ensure a correct global behavior, but also interprocedural analysis, as the queue datatype is very similar to the stack datatype.

We focused last year on the case of Communicating Finite-State Machines (CFSM) [67], a model where the values sent into FIFO queues belongs to bounded datatypes.

This year, we have generalized this approach to infinite-state communicating systems, where both processes and values contained in FIFO queues are infinite-state. We propose in [24] a new abstract domain for languages on infinite alphabets, which acts as a functor taking an abstract domain for a concrete alphabet and lift it to an abstract domain for words on this alphabet. More precisely, provided an abstract domain for the elements of a set S, denoted as $\wp(S) \longleftarrow A$, we build an abstract domain for words on the alphabet S, denoted as $\wp(S^*) \longleftarrow \mathsf{Reg}(A)$. The abstract representation is based on so-called *lattice automata*, which are finite automata labelled by the elements of an atomic lattice, that recognize words on atoms of this lattice. We have defined a normal form, standard language operations and a widening operator for these automata. We have applied this abstract lattice for the verification of symbolic communicating machines, and we have discussed its usefulness for interprocedural analysis.

6.4.3. Automatic Test Generation from Interprocedural Specifications

Participants: C. Constant, B. Jeannet [contact person], T. Jéron.

With the VERTECS team, we have continued our collaboration on model-based testing, using static analysis methods for a precise selection. In [17], we have extended the principles and algorithms of model-based testing for recursive interprocedural specifications that can be modeled by (finite) Push-Down Systems (PDS) Such specifications may be more compact than non-recursive ones and are more expressive. The generated test cases are selected according to a test purpose, a (set of) scenario of interest that one wants to observe during test execution. The test generation method we have proposed is based on program transformations and a coreachability analysis, which allows to decide whether and how the test purpose can still be satisfied. However, despite the possibility to perform an exact analysis, the inability of test cases to inspect their own stack prevents it from using fully the coreachability information. We have analyzed this partial observation problem, its consequences, and we have proposed some solutions to minimize its impact.

6.5. Component-based Construction

Participants: G. Goessler [contact person], P. Fradet, A. Girault, M. Tivoli.

Component-based construction techniques are crucial to overcome the complexity of embedded systems design. However, two major obstacles need to be addressed: the heterogeneous nature of the models, and the lack of results to guarantee correction of the composed system. The heterogeneity of embedded systems comes from the need to integrate components using different models of computation, communication, and execution, on different levels of abstraction and different time scales. The component framework and verification and construction algorithms have to support this heterogeneous nature of the components.

6.5.1. Adapter Synthesis for Synchronous Components

In the context of the ACI Alidecs (see section 8.2.1), we have an ongoing research project on the definition of a language and framework for the construction of safe embedded systems based on synchronous components.

Building a real-time system from existing components introduces several problems, mainly related to compatibility, communication, and QoS issues. We have proposed an approach to automatically synthesize adapters in order to solve black-box integration incompatibilities within a lightweight component model. Adapter synthesis allows the developer to automatically build correct-by-construction systems from third-party components, hence, reducing time-to-market and improving reusability.

A component interface includes a formal description of the interaction protocol of the component with its expected environment. The interface language is expressive enough to specify real-time constraints and controllability of the component actions (ports), as well as the component's activation clock. Based on results from Petri net and supervisory control theory, we have developed and implemented an algorithm which automatically synthesizes deadlock-free bounded-memory adapter components from the interface specification of the components. The generated adapters coordinate the interaction behavior of the components and buffer their communications, in order to avoid deadlocks [26].

We have further formalized the technique, improved separation of concerns by distinguishing between constraints on (local) component time and (global) application time, and generalized the synthesis to adapters working on a sub-clock of the global clock.

6.5.2. Component Refinement

In the context of our work on compositionality and reconfigurability, we recently started studying the issue of component refinement with respect to the existence of a winning strategy (against and adverse environment) for reachability properties. We work on two closely related problems:

(1) In order to find a winning strategy in a huge state space (up to 10^{150} states), we decompose the problem into subtasks that can either be carried out locally on the state space of individual components, or efficiently on the global system. We introduce a property of composability, which guarantees that the locally computed solutions form a solution of the global problem.

(2) The preservation of winning strategies under system reconfiguration (that is, a set of components is replaced with other components), is ensured by a simulation relation, which can be checked locally.

This new work direction is still in progress.

6.5.3. Component Fusion

Given a system of concurrent components communicating through FIFO queues (*i.e.*, a Kahn process network), the technique of network fusion [51] allows to obtain a sequential implementation, thus getting rid of context switching and improving efficiency. We are studying the extension of the component language with non-determinism features (*e.g.*, testing the size of a queue). Introducing non-determinism in Kahn process networks entails some non-trivial problems for component fusion. We have been working on extending the fusion algorithm to fulfill the following requirements: (1) preserve functional (non-confluent) non-determinism, so as to observe the same non-deterministic behavior as in the original component network; (2) eliminate confluent non-determinism as far as possible to improve performance; (3) guarantee fairness. This work is still in progress.

6.6. Aspect-oriented programming

Participants: S. Djoko Djoko, R. Douence, P. Fradet [contact person], A. Girault.

The goal of Aspect-Oriented Programming (AOP) is to isolate aspects (such as security, synchronization, or error handling) which cross-cut the program basic functionality and whose implementation usually yields tangled code. In AOP, such aspects are specified separately and integrated into the program by an automatic transformation process called *weaving*.

Although this new paradigm has great practical potential, it still lacks formalization and undisciplined uses make reasoning on programs very difficult. Our work on AOP addresses these issues by studying foundational issues (semantics, analysis, verification) and by considering domain-specific aspects (availability or fault tolerance aspects) as formal properties.

6.6.1. Aspects preserving properties

Aspect Oriented Programming can arbitrarily distort the semantics of programs. In particular, weaving can invalidate crucial safety and liveness properties of the base program. We have identified categories of aspects that preserve some classes of properties [18]. It is sufficient to check that an aspect belong to a specific category to know which properties will remain satisfied by woven programs.

Our categories of aspects, inspired by Katz's, comprise observers, aborters and confiners. Observers do not modify the base program's state and control-flow (*e.g.*, persistence, profiling and debugging aspects). Aborters are observers which may also abort executions (*e.g.*, security aspects). Confiners only ensure that executions remain in the reachable states of the base program (optimization or fault-tolerance aspects). These categories are defined precisely based on a language independent abstract semantics framework [18].

The corresponding classes of properties are defined as subsets of LTL for deterministic programs (CTL* for non-deterministic ones). We have formally proved that, for any program, the weaving of any aspect in a category preserves any property in the related class.

We are currently working on the design, for each category of aspects, of a domain-specific aspect language ensuring that any aspect written is that language belongs to the corresponding category (*e.g.*, observers, aborters, etc). These languages would guarantee the preservation of key properties by construction.

This work is the central topic of Simplice Djoko Djoko's PhD thesis. It is conducted within the Formal Methods Lab of the network of excellence AOSD-Europe (see section 8.3.2) in collaboration with Rémi Douence from the OBASCO project team at École des Mines de Nantes.

6.6.2. *Resource management and aspects of availability*

We have studied the use of aspect-oriented programming for resource management with the aim of enforcing availability properties [21]. Our technique permits to keep resource management and availability issues separate from the rest of the system.

We propose a domain-specific aspect language in order to prevent denials of service caused by resource management (*e.g.*, starvation, deadlocks, etc.). Availability aspects specify time or frequency limits in the allocation of resources. They can be seen as formal temporal properties on execution traces that specify availability policies. For example, a constraint may be that a program does not retain a resource more than n seconds or that it does not allocate the resource R_1 less than n seconds after it has released the resource R_2.

The semantics of base programs and aspects are expressed as *timed automata*. The automaton representing a program specifies a superset of all possible (timed) execution traces whereas the automaton representing an aspect specifies a set of desired/allowed (timed) execution traces. Weaving can be seen as a *product* of two timed automata (*i.e.*, the intersection of execution traces) which restricts the execution of the base program to the behaviors allowed by the aspect.

The main advantage of such a more formal approach is two-fold:

- aspects are expressed at a higher-level and the semantic impact of weaving is kept under control;
- model checking tools can be used to optimize weaving and verify the enforcement of general availability properties.

This research was part of Stéphane Hong Tuan Ha's PhD thesis from the LANDE project team at IRISA/INRIA-Rennes and supervised by Pascal Fradet.

6.6.3. *Fault tolerance aspects for real-time software*

Here, our objective is to design an aspect language for specifying fault tolerance as well as efficient techniques based on static analysis, program transformation and/or instrumentation to weave them into real-time programs.

As a first step, we have studied the implementation of specific fault tolerance techniques in real-time embedded systems using program transformation [11]. The fault-intolerant initial system consists of a set of independent periodic tasks scheduled onto a set of fail-silent processors. The tasks are automatically transformed such that, assuming the availability of an additional spare processor, the resulting system tolerates one failure at a time. Failure detection is implemented using heartbeating, and failure masking using checkpointing and roll-back. These techniques are described and implemented by automatic program transformations of the tasks' source programs. The proposed formal approach to fault tolerance by program transformation highlights the benefits of separation of concerns.

The second step, is to design an aspect language allowing users to specify and tune a wider range of fault tolerance techniques. For example, the user may want to use checkpointing, code or data replication at different places of the same program. For checkpointing, the user may also want to specify the subset of variables which must be saved. The definition of an aspect language to specify such choices is under completion.

This line of research is related to the ALIDECS project (see section 8.2.1).

6.7. Other results

6.7.1. Programming models and calculi

Participant: P. Fradet.

We have been interested for a long time in formal calculi in order to study programming language issues in the simplest possible setting. We present here work within the λ-calculus (compilation of higher-order sequential languages) and the γ-calculus (higher-order parallel and non-deterministic programming).

6.7.1.1. λ-calculus and the Krivine abstract machine

The Krivine machine is a simple and natural implementation of the call-by-name λ-calculus. While its original description has remained unpublished, this machine has served as a basis for many variants, extensions and theoretical studies. We have presented the Krivine machine and some well-known variants in a common framework [14]. We have characterized the essence of the Krivine machine and have located it in the design space of functional language implementations. This work is based on the framework that we had previously developed for the systematic study of functional language implementations [50].

This is joint work with Rémi Douence from the OBASCO project team (École des Mines de Nantes).

6.7.1.2. γ-calculus and higher-order chemical programming

The chemical reaction metaphor describes computation in terms of a chemical solution in which molecules (representing data) interact freely according to reaction rules. Formally, chemical programs can be represented as associative-commutative rewritings (reactions) of multisets (chemical solutions).

This model of computation is well-suited to the specification of complex computing infrastructures. In particular, the orderless interactions between elements that occur in large parallel or open systems are naturally expressed as reaction rules.

We have been working on the application of HOCL to the programming of distributed applications, in particular to autonomic systems [12]. We have shown that autonomicity features (*e.g.*, self-healing, self-protection, self-optimization, etc.) are naturally expressed as reaction rules.

This work is conducted in collaboration with Jean-Pierre Banâtre and Yann Radenac from the PARIS project team at IRISA. It was the central topic of Yann Radenac's PhD thesis. This line of research is related to the AUTOCHEM project (see section 8.2.1) starting this year.

6.7.2. Component-based modeling and analysis of genetic networks

Participants: G. Goessler [contact person], A. Richard.

Genetic regulatory networks usually encompass a large number of genes, proteins, and metabolites. Being able to model and analyze its behavior is crucial for understanding the interactions between the proteins, and their functions. There has been a wide variety of modeling approaches, including the influential early work of [77] based on logical equations, and [55] based on differential equations. However, simulation and verification of the continuous model are expensive, and many properties are not even decidable in this framework. The approach of [79] based on the approximation of nonlinear models by piecewise linear differential inclusions, uses a discrete abstraction preserving the qualitative dynamics of networks. As [79] approximates the continuous behavior with a monolithic discrete transition system, it still suffers from state space explosion. This problem has been addressed with the component-based approach of [56] where the discrete abstraction is constructed and analyzed modularly, allowing to deal with complex, high-dimensional systems. We have further improved this technique by allowing for a more precise, conservative abstraction.

Using the same approach, we are currently studying, in cooperation with H. de Jong (Helix) and G. Batt (Contraintes), the definition of a symbolic representation of the network behavior as a compact exchange format between the Genetic Network Analyzer (GNA) developed in the Helix group, and the model cheker CADP developed by Vasy.

The lack of numerical values for the parameters characterizing the interactions of a genetic regulatory network makes classical numerical analysis techniques difficult to apply. The approach of [79] defines a discrete abstraction preserving the qualitative dynamics of networks for wide ranges of parameter values. We have developed, in cooperation with H. de Jong, a novel algorithm to enumerate all classes of parameter values of an incompletely specified network. This technique has been implemented and applied to the analysis of a model of a network controlling the stress response of bacteria, and has allowed to uncover a shortcoming in the model.

6.7.3. Interactions Between Law and Information and Communication Sciences

Participant: D. Le Métayer [contact person].

Daniel Le Métayer is initiating a new activity (which is to become an independent action in the short term) on the interactions between ICT (Information and Communication Technologies) and law. The motivation for this activity is the observation that the impact of ICT on the every day life of most individuals raises new challenges which cannot be tackled by a purely technological approach. Our position is that the first step for a fruitful and useful exploration of the relationship between the legal and technical dimensions is the definition of a formal framework for expressing the notions at hand, understanding them without ambiguity, and eventually relating or combining them. The first application of this approach, which is conducted within the PRIAM ARC, concerns privacy protection in the ambient intelligence context. Privacy is a complex issue, especially in the context of ambient intelligence, both from the legal and the technical perspective:

- The very definition of privacy is far from obvious since it is by essence subjective and based on a fuzzy notion of boundary or private sphere. In particular, these boundaries are blurred in the ambient intelligence landscape. In addition, the legal framework, which has to reflect the social expectations, needs to be revisited to account for the new possibilities offered by the technology.
- An ambient computing infrastructure is by nature heterogeneous and dynamic, with new nodes, of different natures and belonging to potentially unknown mistrusting users connecting to the network and being able to communicate in a spontaneous way. Last but not least, the smart objects can be tiny, inexpensive, devices with limited resources (chips on clothes, banknotes, etc.). It is thus difficult to rely on these to implement complex privacy policies.

The PRIAM project emphasizes the design of privacy policies that are amenable to both a formal description and a realistic implementation in the ambient world. The techniques under study are based on a combination of a priori controls (*e.g.*, access controls), which are the most conservative, and a posteriori controls (*e.g.*, audits) which may be easier to enforce on resource-constrained devices.

6.7.4. Control for data-parallel systems

Participant: E. Rutten.

Data intensive computing is increasingly getting high importance in a wide range of scientific and engineering domains. Such systems manipulate large amounts of data; so high performance, scalability and throughput are important requirements. Reconfigurability is another interesting feature because it makes the systems flexible enough to be adapted to various environment and resource constraints. The GASPARD2 [32] development framework aims at proposing a solution to the design of data intensive applications in general, and high-performance embedded system-on-chip (SoCs) in particular.

We have proposed a synchronous model of GASPARD2, in order to bridge the gap between GASPARD2 and analysis and verification tools of the synchronous technology so that formal validation is favored [32].

The automation of the transformations is implemented within an MDE framework [27].

We extend GASPARD2, by adding reactive control features based on finite state machines [15], [33], and are integrating this extension in the synchronous model.

This work is conducted in cooperation with the DART project at UR Futurs in Lille.

[32] http://www.lifl.fr/west/gaspard

7. Contracts and Grants with Industry

7.1. Pôle de compétitivité Minalogic/EMSOC

In the context of the pôle de compétitivité EMSOC/Minalogic, we participate in the four-year project OpenTLM on analysis of systems-on-chip modeled at the transaction level in SystemC [60]. We intend to develop methods for abstraction, and interprocedural and compositional analysis of SystemC models. Two PhD students will be hired on these topics.

7.2. DCN

With the INRIA project team MOAIS and the ProBayes start-up, we have signed a contract with DCN. DCN is a French company based in Toulon that builds warships. We will work on a R&D project aimed at improving the defense embedded software of their next generation warships.

8. Other Grants and Activities

8.1. Regional actions

8.1.1. Regional cluster ISLE

We participate in the regional cluster ISLE ("Informatique, Systèmes et Logiciels Embarqués") of the Région Rhône-Alpes, which funds the PhD of Mouaiad Alras (see Section 6.2).

8.2. National actions

8.2.1. ACI "Sécurité & Informatique" Alidecs: integrated development environment for safe embedded components

Participants: P. Fradet, A. Girault, G. Goessler.

The objective of the ALIDECS project[33] is to study an integrated development environment for the construction and use of safe embedded components. The consortium includes LRI (Orsay), INRIA (Rhône-Alpes and Sophia Antipolis), VERIMAG (Grenoble) and LAMI (Evry). We have proposed weaving-like techniques for enforcing fault tolerance properties to reactive systems. We have also studied an approach to automatically synthesize adaptors in order to assemble off-the-shelf real-time components. The project has ended in October 2007.

8.2.2. ANR AutoCHEM

The AUTOCHEM aims at investigating and exploring the use of rule-based languages such as HOCL (see section 6.7.1) to program complex computing infrastructures such as Grids and real-time deeply-embedded systems. The consortium includes IRISA (PARIS project team, Rennes), INRIA-Rhône-Alpes (Pop Art project team, Montbonnot), IBISC (CNRS/Université d'Evry) and CEA List (Saclay). The project has started at the end of 2007.

8.2.3. ACI "Sécurité et informatique" Apron: analysis of numerical programs

Participant: B. Jeannet.

The APRON (Analyse de PROgrammes Numériques) project (http://www.cri.ensmp.fr/apron/) [2004-2006] involves the team Analyses, transformations et instrumentations de programmes (Centre de Recherche en Informatique, École des Mines de Paris [coordinator]), the Synchrone team (Verimag, Grenoble), the VERTECS and now POP-ART project (INRIA), and the team Sémantique, preuves et interprétation abstraite (École Polytechnique, Palaiseau).

[33] http://www-verimag.imag.fr/SYNCHRONE/alidecs/

The focus of this project is on the theory of numerical abstract domains and their application to the static analysis of the numerical variables of a program. The first, theoretical goal of the project is to advance the research in numerical abstract domains. The second, more practical goal, is to mature the field by bringing together five actors to define their needs, and then design and implement a common software platform suited for a broad range of static analysis applications. This project has lead to 29 publications in international conferences, workshops, and journals, four technical reports, four Ph.D. theses, and the design and implementation of the APRON numerical abstract domain library (5.4). The project started in October 2004 and ended in October 2007.

In 2007, most of the effort of POP ART was spent on the implementation and the dissemination of the APRON library.

8.2.4. ARC "PRIAM

Participant: D. Le Métayer.

The goal of the PRIAM ARC is to put forward technological and legal solutions to enhance privacy protection in the ambient intelligence context. The project has started in January 2007. The partners are INRIA (POP ART, ACES, ARES), the law faculty of Saint-Etienne (Joël Moret-Bailly) and the university of Twente (Sandro Etalle).

8.2.5. CNRS RTP 21: fault tolerance

We are collaborating to this RTP entitled *Sûreté de fonctionnement des systèmes informatiques complexes ouverts*[34].

8.2.6. Collaborations inside Inria

- AOSTE at INRIA-Rocquencourt is working with us on fault tolerant heuristics for their software SYNDEX.
- VERTECS at IRISA/INRIA-Rennes is working with us on applications of discrete controller synthesis, and in particular on the tool SIGALI.
- P. Fradet cooperates with J.-P. Banâtre, T. Priol and Y. Radenac (PARIS, IRISA/INRIA-Rennes) and with R. Douence and M. Südholt (OBASCO, Ecole des Mines de Nantes).
- A. Girault cooperates with the MOAIS project (UR Rhône-Alpes) on multi-criteria scheduling. In particular, we have a common industrial contract with DCN. A. Girault cooperates also with the VERIMAG lab on model-based design and a compilation tool chain from SIMULINK to distributed platforms, and with the DEMON team of LRI (Orsay) on the distribution of higher-order synchronous data-flow programs.
- G. Goessler cooperates with H. de Jong (HELIX project, UR Rhône-Alpes) and F. Lang (VASY project, UR Rhône-Alpes).
- B. Jeannet cooperates with T. Le Gall (VERTECS, IRISA/INRIA-Rennes) on the analysis of communicating systems, and with C. Constant, T. Jéron and F. Ployette (VERTECS, IRISA/INRIA-Rennes) on test generation.
- D. Le Métayer cooperates with the ARES (Stéphane Ubéda, Marine Minier, Frédéric Le Mouël) and ACES (Ciaran Bryce) Inria project-teams within the PRIAM ARC.
- E. Rutten is working with the DART project at UR Futurs in Lille, on the synchronous modelling of massively parallel application, and the introduction of control and mode automata in the GASPARD framework.

8.2.7. Cooperations with other laboratories

- P. Fradet cooperates with S. Hong Tuan Ha (CEA Saclay).

[34] http://www.laas.fr/RTP21-SdF

- A. Girault cooperates with X. Nicollin (VERIMAG), M. Pouzet (LRI, University of Paris VI), D. Trystram and É. Saule from (ID-IMAG), and C. Dima (Université of Paris XII).
- G. Goessler cooperates with J. Sifakis and S. Graf (VERIMAG) and M. Majster-Cederbaum (University of Mannheim, Germany).
- B. Jeannet cooperates with N. Halbwachs and L. Gonnord (VERIMAG) and A. Miné (ENS Paris) on the static analysis of numerical variables.
- D. Le Métayer cooperates with faculté de droit de Saint-Etienne, université de Twente within the PRIAM ARC.
- E. Rutten cooperates with H. Alla (GIPSA).

8.3. European actions

8.3.1. Artist II European IST network of Excellence

ARTIST II is a European Network of Excellence on embedded system design[35]. Its goal is to establish Embedded Systems Design as a discipline, combining expertises from electrical engineering, computer science, applied mathematics, and control theory. We collaborate as a core partner within the Real Time Components cluster, led by A. Benveniste (INRIA Rennes) and B. Jonsson (Uppsala University). A. Girault is the administrator of ARTIST II for INRIA.

8.3.2. AOSD European IST network of Excellence

AOSD-Europe is the European network of excellence on Aspect-Oriented Software Development. It lasts 4 years (September 2004-August 2008) and includes nine major academic institutions and two major industrial partners from UK, Germany, The Netherlands, France, Belgium, Ireland, Spain and Israel. We collaborate in the formal methods lab with OBASCO-INRIA, Technion (Israel), and Twente (The Netherlands).

8.3.3. Interlink Action

InterLink is a coordination action funded by the European Commission. It aims at advancing Europes knowledge in a number of critical Information and Communication Science and Technologies areas. P. Fradet has participated to the first InterLink workshop on software intensive systems and new computing paradigms, May 2007, Eze.

8.4. Actions internationales

8.4.1. CMCU Tunisia

We have a cooperation in the framework of CMCU (*Comité Mixte pour la Coopération Universitaire*), on the topic of analysis and verification of the safety of safety-critical systems, with ENSI (*Ecole Nationale des Sciences de l'Informatique*) at La Manouba in Tunisia. The other french partner is GIPSA (team of Hassana Alla).

9. Dissemination

9.1. Scientific community

- P. Fradet has participated in the program committee of FOAL'07 (*Foundations of Aspect-Oriented Languages Workshop*). He has organized (with R. Douence) JFDLA 2007 in Toulouse (3ème Journée Francophone sur le Développement de Logiciels Par Aspects) http://pop-art.inrialpes.fr/~jfdlpa07/. He has given a course with Jean-Pierre Banâtre on Chemical Programming at *Ecole des Jeunes Chercheurs en Programmation*, Dinard, juin 2007. He was member of the PhD committee of David Stauch (INPG, November 2007).

[35] http://www.artist-embedded.org/FP6

- A. Girault organized the SYNCHRON'06 Workshop[36] in L'Alpe-d'Huez (France), co-chaired the programme committe of the FMGALS'07 Workshop[37] in Nice (France), and SLA++P08, the workshop on Model-driven High-level Programming of Embedded Systems, co-located with ETAPS08. and served in the programme committee of the MSR'07 and DATE'08 Conference[38]. He was PhD assessor for the jury of Julien Boucaron (University of Nice).
- Daniel Le Métayer has participated in the program committees of FASE'08 (*Fundamental Aspects of Software Engineering*) and ARES/APE08.
- E. Rutten is co-editor of the special issue of Discrete Event Dynamical Systems (jDEDS) on Control and Modeling of Reactive Systems [10]. He is a co-chair of SLA++P08[39], the workshop on Model-driven High-level Programming of Embedded Systems, co-located with ETAPS08. He participated in the program committee of MSR07. He was member of the PhD committee of A. Medina (ENS Cachan). He is a member of the *Commission de spécialistes* (recruiting commitee) at the Universities of Brest (UBO) and Lille-1.

9.2. Teaching

9.2.1. Courses

- Alain Girault: *Algorithmics and programming in Java*, 26h, INPG Telecom Department.
- Gregor Goessler: *Software Engineering and Compilation project*, 2nd year engineering, 55h, INPG / ENSIMAG.
- Alain Girault and Pascal Raymond: *Synchronous programming*, 28h, Master of Science, Université Joseph Fourier.
- Daniel Le Métayer: *Systematic security analysis*, at the Ecole des Mines de Nantes, November 2006. 3h.

9.2.2. Advising

PhDs:

- Stéphane Hong Tuan Ha, advised by Pascal Fradet, since 9/2002, PhD in computer science, University of Rennes I. Thesis defended on January 30, 2007.
- Yann Radenac, co-advised by P. Fradet (with J.-P. Banâtre, IRISA), since 9/2003, PhD in computer science, University of Rennes I. Thesis defended on April 18, 2007.
- Gwenaël Delaval, co-advised by Alain Girault (with M. Pouzet, LRI Orsay), since 9/2004. PhD in computer science, INPG.
- Tristan Le Gall, co-advised by Bertrand Jeannet (with T. Jéron, VERTECS IRISA) since 9/2004. PhD in computer science, University of Rennes I.
- Simplice Djoko Djoko, co-advised by P. Fradet (with R. Douence, OBASCO, Ecole des Mines de Nantes), since 9/2005, PhD in computer science, University of Nantes.
- Huafeng Yu, co-advised by E. Rutten (with J.-L. Dekeyser, LIFL/INRIA Futurs Lille), since 10/2005. PhD in computer science, University of Lille 1.
- Mouaiad Alras, co-advised by Alain Girault (with P. Raymond, VERIMAG Grenoble), since 10/2006, PhD in computer science, UJF, Grenoble.
- Camille Constant, co-advised by Bertrand Jeannet (with T. Jéron, VERTECS IRISA) since 9/2006. PhD in computer science, University of Rennes I.

[36] SYNCHRON'06: http://www.inrialpes.fr/Synchron06
[37] FMGALS'07: http://www.inrialpes.fr/Fmgals07
[38] DATE'08: http://www.date-conference.com
[39] SLA++P08: http://www.inrialpes.fr/sla++p08/

- Gérald Vaisman, co-advised by Alain Girault (with P.-F. Dutot, MOAIS UR Rhône-Alpes), since 10/2006, PhD in computer science, INPG.

Masters:

- Xing Lu, in 2006/2007. Master of Science in computer science (M2R), UJF, Grenoble.

10. Bibliography

Major publications by the team in recent years

[1] K. ALTISEN, A. CLODIC, F. MARANINCHI, É. RUTTEN. *Using Controller-Synthesis Techniques to Build Property-Enforcing Layers*, in "Proceedings of the European Symposium on Programming, ESOP'03", Lecture Notes in Computer Science (LNCS), n^{o} 2618, Springer Verlag, April 2003, p. 174–188.

[2] K. ALTISEN, G. GOESSLER, J. SIFAKIS. *Scheduler Modeling Based on the Controller Synthesis Paradigm*, in "Journal of Real-Time Systems, special issue on "control-theoretical approaches to real-time computing"", vol. 23, n^{o} 1/2, 7-9 2002, p. 55–84.

[3] I. ASSAYAD, A. GIRAULT, H. KALLA. *A Bi-Criteria Scheduling Heuristics for Distributed Embedded Systems Under Reliability and Real-Time Constraints*, in "International Conference on Dependable Systems and Networks, DSN'04, Firenze, Italy", IEEE, June 2004, p. 347–356.

[4] P. CASPI, A. GIRAULT, D. PILAUD. *Automatic Distribution of Reactive Systems for Asynchronous Networks of Processors*, in "IEEE Trans. on Software Engineering", vol. 25, n^{o} 3, May 1999, p. 416–427.

[5] T. COLCOMBET, P. FRADET. *Enforcing trace properties by program transformation*, in "Proc. of Principles of Programming Languages, Boston", ACM Press, January 2000, p. 54-66.

[6] P. FRADET. *Approches langages pour la conception et la mise en œuvre de programmes*, Habilitation thesis, Université de Rennes 1, November 2000.

[7] P. FRADET, S. HONG TUAN HA. *Network Fusion*, in "Proceedings of Asian Symposium on Programming Languages and Systems (APLAS'04)", LNCS, vol. 3302, Springer-Verlag, November 2004, p. 21–40.

[8] A. GIRAULT, H. KALLA, M. SIGHIREANU, Y. SOREL. *An Algorithm for Automatically Obtaining Distributed and Fault-Tolerant Static Schedules*, in "International Conference on Dependable Systems and Networks, DSN'03, San-Francisco (CA), USA", IEEE, June 2003.

[9] G. GOESSLER, J. SIFAKIS. *Priority Systems*, in "proc. FMCO'03", F. DE BOER, M. BONSANGUE, S. GRAF, W.-P. DE ROEVER (editors), LNCS, vol. 3188, Springer-Verlag, 2004, p. 314-329.

Year Publications

Books and Monographs

[10] H. ALLA, R. BOEL, E. RUTTEN (editors). *Modelling and Control of Reactive Systems*, Special issue of the journal of Discrete Event Dynamic System, Volume 17, number 2, May 2007, http://springerlink.metapress.com/content/1573-7594/.

Articles in refereed journals and book chapters

[11] T. AYAV, P. FRADET, A. GIRAULT. *Implementing Fault-Tolerance by Automatic Program Transformations*, in "ACM Transactions on Embedded Computing Systems", (to appear), 2007.

[12] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *Programming Self-Organizing Systems with the Higher-Order Chemical Language*, in "International Journal of Unconventional Computing", vol. 3, n° 3, 2007, p. 161–177.

[13] G. DELAVAL, E. RUTTEN. *A domain-specific language for multi-task systems, applying discrete controller synthesisd*, in "Journal on Embedded Systems (special issue on Synchronous Paradigm in Embedded Systems)", vol. 2007, n° 84192, January 2007, 17, http://www.hindawi.com/GetArticle.aspx?doi=10.1155/2007/84192.

[14] R. DOUENCE, P. FRADET. *The next 700 Krivine Machines*, in "Higher-Order and Symbolic Computation", vol. 20, n° 3, 2007.

[15] O. LABBANI, JEAN-LUC. DEKEYSER, P. BOULET, E. RUTTEN. *UML2 profile for modelling controlled data parallel applications*, in "Advances in Design and Specification Languages for Embedded Systems", S. A. HUSS (editor), ISBN: 978-1-4020-6147-9, Springer Verlag, September 2007.

Publications in Conferences and Workshops

[16] C. BRYCE, M. DEKKER, S. ETALLE, D. LE MÉTAYER, F. LE MOUËL, M. MINIER, J. MORET-BAILLY, S. UBÉDA. *Ubiquitous privacy protection*, in "5th International Workshop on Privacy in UbiComp (UbiPriv'07)", 2007.

[17] C. CONSTANT, B. JEANNET, T. JÉRON. *Automatic Test Generation from Interprocedural Specifications*, in "Testing of Communicating Systems and Formal Approaches to Testing of Software, TESTCOM/FATES'07", LNCS, vol. 4581, July 2007, http://pop-art.inrialpes.fr/people/bjeannet/publications/testcom07.ps.gz.

[18] S. DJOKO DJOKO, R. DOUENCE, P. FRADET. *Aspects Preserving Properties*, in "Proc. of the ACM SIGPLAN 2008 Symposium on Partial Evaluation and Program Manipulation (PEPM'08)", To appear, ACM, january 2008.

[19] E. DUMITRESCU, A. GIRAULT, H. MARCHAND, É. RUTTEN. *Optimal discrete controller synthesis for modeling of fault-tolerant distributed systems*, in "Proceedings of the 1st IFAC Workshop on Dependable Control of Discrete Systems, DCDS'07, Paris - Cachan, France, June 13-15, 2007", 2007.

[20] E. DUMITRESCU, A. GIRAULT, H. MARCHAND, É. RUTTEN. *Synthèse optimale de contrôleurs discrets pour des systèmes distribués tolérants aux fautes*, in "Actes du 6ème Colloque Francophone sur la Modélisation des Systèmes Réactifs, MSR'07 Lyon, France, 17-19 octobre 2007", 2007.

[21] P. FRADET, S. HONG TUAN HA. *Aspects of Availability*, in "Proc. of the Sixth International Conference on Generative Programming and Component Engineering (GPCE'07)", ACM, october 2007, p. 165–174.

[22] G. GOESSLER, S. GRAF, M. MAJSTER-CEDERBAUM, M. MARTENS, J. SIFAKIS. *An Approach to Modelling and Verification of Component Based Systems*, in "proc. SOFSEM'07", LNCS, Springer-Verlag, 2007.

[23] G. GOESSLER, S. GRAF, M. MAJSTER-CEDERBAUM, M. MARTENS, J. SIFAKIS. *Ensuring Properties of Interaction Systems by Construction*, in "Program Analysis and Compilation, Theory and Practice", T. REPS, M. SAGIV, J. BAUER (editors), LNCS, vol. 4444, 2007, p. 201-224.

[24] T. LE GALL, B. JEANNET. *Lattice automata: a representation of languages over an infinite alphabet, and some applications to verification*, in "Static Analysis Symposium, SAS'07", LNCS, vol. 4634, August 2007, http://pop-art.inrialpes.fr/people/bjeannet/publications/sas07.ps.gz.

[25] D. LE MÉTAYER. *IT security analysis: best practices and formal approaches*, in "proc. Foundations of Security Analysis and Design IV (FOSAD)", LNCS, vol. 4677, Springer, 2007.

[26] M. TIVOLI, P. FRADET, A. GIRAULT, G. GOESSLER. *Adaptor Synthesis for Real-Time Components*, in "Tools and Algorithms for the Construction and Analysis of Systems, 13th International Conference (TACAS'07)", Lecture Notes in Computer Science, vol. 4424, Springer, 2007, p. 185-200.

[27] H. YU, A. GAMATIÉ, E. RUTTEN. *Model Transformations from a Data Parallel Formalism towards Synchronous Languages*, in "Proceedings of the Forum on specification and Design Languages, FDL'07, Barcelona, Spain September 18-20, 2007", 2007.

Internal Reports

[28] C. CONSTANT, B. JEANNET, T. JÉRON. *Automatic Test Generation from Interprocedural Specifications*, Technical report, n^{o} 1835, IRISA, March 2007, http://www.irisa.fr/vertecs/Publis/Ps/PI-1835.pdf.

[29] E. DUMITRESCU, A. GIRAULT, H. MARCHAND, E. RUTTEN. *Optimal discrete controller synthesis for the modeling of fault-tolerant distributed systems*, Research Report, n^{o} 6137, INRIA, March 2007, http://hal.inria.fr/inria-00134550.

[30] A. GIRAULT, H. KALLA. *Revisiting the Bicriteria (length,reliability) Multiprocessor Static Scheduling Problem*, Research Report, n^{o} 6319, INRIA, Grenoble, France, October 2007, http://hal.inria.fr/inria-00177117/.

[31] T. LE GALL, B. JEANNET. *Analysis of Communicating Infinite State Machines using Lattice Automata*, Technical report, n^{o} 1839, IRISA, March 2007, http://www.irisa.fr/vertecs/Publis/Ps/PI-1839.pdf.

[32] H. YU, A. GAMATIÉ, E. RUTTEN, J.-L. DEKEYSER. *Model Transformations from a Data Parallel Formalism towards Synchronous Languages*, Research Report, n^{o} 6291, INRIA, September 2007, http://hal.inria.fr/inria-00172302.

Miscellaneous

[33] O. LABBANI, É. RUTTEN, J.-L. DEKEYSER. *Safe Design Methodology for an Intelligent Cruise Control System with GPS*, 2007, IEEE Intelligent Transportation Systems Society Newsletter, vol. 8, nr. 4, pp. 16-23, december 2006.

References in notes

[34] A. ARNOLD. *Systèmes de transitions finis et sémantique des processus communicants*, Masson, 1992.

[35] E. ASARIN, O. BOURNEZ, T. DANG, O. MALER, A. PNUELI. *Effective Synthesis of Switching Controllers of Linear Systems*, in "Proceedings of the IEEE", vol. 88, 2000, p. 1011–1025.

[36] R. BAGNARA, E. RICCI, E. ZAFFANELLA, P. M. HILL. *Possibly not Closed Convex Polyhedra and the Parma Polyhedra Library*, in "Static Analysis Sympisum, SAS'02", LNCS, vol. 2477, 2002.

[37] J.-R. BEAUVAIS, E. RUTTEN, T. GAUTIER, R. HOUDEBINE, P. LE GUERNIC, YAN-MEI. TANG. *Modelling Statecharts and Activity Charts as Signal Equations*, in "ACM Transactions on Software Engineering and Methodology", vol. 10, n^{o} 4, October 2001, p. 397–451.

[38] J.-J. BORRELLY, E. COSTE MANIÈRE, B. ESPIAU, K. KAPELLOS, R. PISSARD-GIBOLLET, D. SIMON, N. TURRO. *The Orccad Architecture*, in "International Journal on Robotic Research", vol. 17, n^{o} 4, 1998, p. 338–359.

[39] R. BRYANT. *Graph-based algorithms for boolean function manipulation*, in "IEEE Transactions on Computers", vol. C-35, n^{o} 8, 1986, p. 677–692.

[40] CEI (COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE). *Norme Internationale – Automates programmables : Langages de programmation*, Technical report, n^{o} IEC 1131 partie 3, CEI/IEC (International Electrotechnical Commission), 1993.

[41] P. CASPI, A. CURIC, A. MAIGNAN, C. SOFRONIS, S. TRIPAKIS, P. NIEBERT. *From Simulink to Scade/Lustre to TTA: A Layered Approach for Distributed Embedded Applications*, in "International Conference on Languages, Compilers, and Tools for Embedded Systems, LCTES'03, San Diego (CA), USA", ACM, June 2003, p. 153–162.

[42] P. CASPI, M. POUZET. *Synchronous Kahn networks*, in "International Conference on Functional Programming, ICFP'96, Philadelphia (PA), USA", ACM, 1996, p. 226–238.

[43] C. CASSANDRAS, S. LAFORTUNE. *Introduction to Discrete Event Systems*, Kluwer, 1999.

[44] D. CHASE, M. WEGMAN, F. ZADECK. *Analysis of Pointers and Structures*, in "Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation", ACM Press, 1990, p. 296–310.

[45] E. CLARKE, E. EMERSON, A. 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, 1986, p. 244-263.

[46] D. CLARKE, T. JÉRON, V. RUSU, E. ZINOVIEVA. *STG: a Symbolic Test Generation tool*, in "(Tool paper) Tools and Algorithms for the Construction and Analysis of Systems (TACAS'02)", LNCS, vol. 2280, 2002.

[47] P. COUSOT, R. COUSOT. *Abstract Interpretation and Application to Logic Programs*, in "Journal of Logic Programming", vol. 13, n^{o} 2–3, 1992, p. 103–179.

[48] P. COUSOT, N. HALBWACHS. *Automatic discovery of linear restraints among variables of a program*, in "5th ACM Symposium on Principles of Programming Languages, POPL'78, Tucson (Arizona)", January 1978.

[49] P. D'ARGENIO, B. JEANNET, H. JENSEN, K. LARSEN. *Reduction and Refinement Strategies for Probabilistic Analysis*, in "Process Algebra and Probabilistic Methods - Performance Modelling and Verification, PAPM-PROBMIV'02, Copenhagen (Denmark)", LNCS, vol. 2399, July 2002.

[50] R. DOUENCE, P. FRADET. *A Systematic Study of Functional Language Implementations*, in "ACM Transactions on Programming Languages and Systems", vol. 20, n^{o} 2, 1998, p. 344–387.

[51] P. FRADET, S. HONG TUAN HA. *Network Fusion*, in "proc. APLAS'04", LNCS, vol. 3302, Springer-Verlag, 2004, p. 21-40.

[52] F. GAUCHER, E. JAHIER, B. JEANNET, F. MARANINCHI. *Automatic State Reaching for Debugging Reactive Programs*, in "5th Int. Workshop on Automated and Algorithmic Debugging, AADEBUG'03", September 2003.

[53] A. GIRAULT. *System-Level Design of Fault-Tolerant Embedded Systems*, vol. 67, October 2006.

[54] A. GIRAULT, H. KALLA, Y. SOREL. *Transient Processor/Bus Fault Tolerance for Embedded Systems*, in "IFIP Working Conference on Distributed and Parallel Embedded Systems, DIPES'06, Braga, Portugal", Springer, October 2006, p. 135–144.

[55] L. GLASS, S. KAUFFMAN. *The logical analysis of continuous, non-linear biochemical control networks*, in "Journal of Theoretical Biology", vol. 39, n^{o} 1, 1973, p. 103-129.

[56] G. GOESSLER. *Compositional Reachability Analysis of Genetic Networks*, in "CMSB'06", C. PRIAMI (editor), LNBI, vol. 4210, Springer, 2006, p. 212-226.

[57] G. GOESSLER, J. SIFAKIS. *Priority Systems*, in "proc. FMCO'03", F. DE BOER, M. BONSANGUE, S. GRAF, W.-P. DE ROEVER (editors), LNCS, vol. 3188, Springer-Verlag, 2004, p. 314-329.

[58] G. GOESSLER, J. SIFAKIS. *Composition for Component-based Modeling*, in "Science of Computer Programming", vol. 55, n^{o} 1-3, 2005, p. 161-183.

[59] D. GOPAN, T. W. REPS. *Guided Static Analysis*, in "Static Analysis Symposium, SAS'07", LNCS, vol. 4634, August 2007.

[60] T. GRÖTKER, S. LIAO, G. MARTIN, S. SWAN. *System Design with SystemC*, Kluwer, 2002.

[61] N. HALBWACHS. *Synchronous Programming of Reactive Systems*, Kluwer, 1993.

[62] N. HALBWACHS. *Synchronous Programming of Reactive Systems – a Tutorial and Commented Bibliography*, in "Proc. of the Int. Conf. on Computer-Aided Verification, CAV'98, Vancouver, Canada", LNCS Vol. 1427, Springer-Verlag, 1998.

[63] D. HAREL. *Statecharts: A Visual Formalism for Complex Systems*, in "Science of Computer Programming", vol. 8, 1987, p. 231-274.

[64] B. JEANNET, P. D'ARGENIO, K. LARSEN. *RAPTURE: A tool for verifying Markov Decision Processes*, in "Tools Day, International Conference on Concurrency Theory, CONCUR'02, Brno (Czech Republic)", Technical Report, Faculty of Informatics at Masaryk University Brno, August 2002.

[65] B. JEANNET. *Dynamic Partitioning In Linear Relation Analysis. Application To The Verification Of Reactive Systems*, in "Formal Methods in System Design", vol. 23, n° 1, July 2003, p. 5–37.

[66] B. JEANNET, T. JÉRON, V. RUSU, E. ZINOVIEVA. *Symbolic Test Selection based on Approximate Analysis*, in "11th Int. Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'05), Edinburgh (UK)", LNCS, vol. 3440, April 2005.

[67] T. LE GALL, B. JEANNET, T. JÉRON. *Verification of Communication Protocols Using Abstract Interpretation of FIFO queues*, in "Algebraic Methodology and Software Technology, AMAST'06", LNCS, vol. 4019, July 2006.

[68] O. MALER, A. PNUELI, J. SIFAKIS. *On the Synthesis of Discrete Controllers for Timed Systems*, in "Proc. of STACS'95", LNCS, vol. 900, Springer Verlag, 1995.

[69] F. MARANINCHI, Y. RÉMOND. *Mode-Automata: a new Domain-Specific Construct for the Development of Safe Critical Systems*, in "Science of Computer Programming", vol. 46, n° 3, March 2003, p. 219-254.

[70] H. MARCHAND, P. BOURNAI, M. LE BORGNE, P. LE GUERNIC. *Synthesis of Discrete-Event Controllers based on the Signal Environment*, in "Discrete Event Dynamical System: Theory and Applications", vol. 10, n° 4, October 2000, p. 325–346.

[71] A. MINÉ. *Symbolic Methods to Enhance the Precision of Numerical Abstract Domains*, in "Verification, Model-Checking and Abstract Interpretation, VMCAI'06", LNCS, vol. 3855, 2002, http://www.di.ens.fr/~mine/publi/article-mine-VMCAI06.pdf.

[72] J.-P. QUEILLE, J. SIFAKIS. *Specification and Verification of Concurrent Systems in CESAR*, in "proc. International Symposium on Programming", LNCS, vol. 137, Springer-Verlag, 1982, p. 337-351.

[73] P. J. RAMADGE, W. M. WONHAM. *Supervisory control of a class of discrete event processes*, in "SIAM J. Control Optim.", vol. 25, n° 1, 1987, p. 206–230.

[74] P. J. RAMADGE, W. M. WONHAM. *The Control of Discrete Event Systems*, in "Proceedings of the IEEE", vol. 77, n° 1, 1989.

[75] S. SHATZ, J.-P. WANG. *Models and Algorithms for Reliability-Oriented Task-Allocation in Redundant Distributed-Computer Systems*, in "IEEE Trans. on Reliability", vol. 38, n° 1, April 1989, p. 16–26.

[76] J.-P. TALPIN, P. JOUVELOT. *Polymorphic Type, Region and Effect Inference*, in "Journal of Functional Programming", vol. 2, n° 3, 1992.

[77] R. THOMAS. *Boolean Formalisation of Genetic Control Circuits*, in "J. Theor. Biol.", vol. 42, 1973, p. 565-583.

[78] S. TRIPAKIS, C. SOFRONIS, P. CASPI, A. CURIC. *Translating Discrete-Time Simulink to Lustre*, in "ACM Trans. on Embedded Computing Systems", vol. 4, n^{o} 4, November 2005, p. 779–818.

[79] H. DE JONG, J.-L. GOUZÉ, C. HERNANDEZ, M. PAGE, T. SARI, J. GEISELMANN. *Qualitative Simulation of Genetic Regulatory Networks Using Piecewise-Linear Models*, in "Bulletin of Mathematical Biology", vol. 66, 2004, p. 301-340.