



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

Activity
R *eport*

2008

Table of contents

1. Team	1
2. Overall Objectives	1
3. Scientific Foundations	2
3.1. Embedded systems and their safe design	2
3.1.1. The safe design of embedded real-time control systems.	2
3.1.2. Models, methods and techniques.	3
3.2. Issues in design automation for complex systems	3
3.2.1. Hard problems	3
3.2.2. Applicative needs	4
3.2.3. Our approach	4
3.3. Main Research Directions	4
3.3.1. Principles	4
3.3.2. Implementations of synchronous programs	5
3.3.3. Automatic generation of correct controllers	5
4. Application Domains	5
4.1. Industrial applications.	5
4.2. Industrial design tools	6
4.3. Current industrial cooperations.	6
5. Software	6
5.1. NBac	6
5.2. Prometheus	7
5.3. Implementations of synchronous programs	7
5.3.1. Code distribution	7
5.3.2. Fault tolerance	7
5.4. APRON library	7
5.4.1. Motivation and Principles.	8
5.4.2. Implementation.	8
5.5. Prototypes	9
5.5.1. Automatic Controller Generation	9
5.5.2. Rapture	9
5.5.3. Libraries for Abstract Interpretation	10
6. New Results	10
6.1. Dependable distributed real-time embedded systems	10
6.1.1. Static multiprocessor scheduling with tradeoff between performance and reliability	10
6.1.2. Static multiprocessor scheduling of tasks with resource constraints	11
6.2. Automatic distribution of synchronous programs	11
6.2.1. Modular distribution and application to discrete controller synthesis	11
6.2.2. Model-based development of fault-tolerant embedded systems, code generation for distributed heterogeneous platforms	12
6.3. Automatic generation of correct controllers	12
6.3.1. Distributed Controller Synthesis using Static Analysis of FIFO channels	12
6.3.2. Domain-specific language for application of discrete controller synthesis	12
6.3.3. Automating the addition of fault tolerance with discrete controller synthesis	13
6.3.4. Model-based control of adaptative systems	13
6.4. Static Analysis and Abstract Interpretation	14
6.4.1. Combining control and data abstraction in the verification of hybrid systems	14
6.4.2. A Relational Approach to Interprocedural Shape Analysis	14
6.4.3. Relational interprocedural analysis of concurrent programs	14
6.5. Component-Based Construction	15

6.5.1.	Specification enforcing refinement for convertibility verification	15
6.5.2.	Compositional strategy mapping	15
6.5.3.	Contract-based design	15
6.6.	Aspect-oriented programming	16
6.6.1.	Aspects preserving properties	16
6.6.2.	Resource management and aspects of availability	16
6.6.3.	Fault tolerance aspects for real-time software	17
6.7.	Other results	17
6.7.1.	Chemical programming	17
6.7.2.	Component-based modeling and reachability analysis of genetic networks	17
6.7.3.	Control for data-parallel systems	18
7.	Contracts and Grants with Industry	18
7.1.	Pôle de compétitivité Minalogic: OpenTLM	18
7.2.	DCNS	18
7.3.	Pôle de compétitivité Minalogic: MIND	19
8.	Other Grants and Activities	19
8.1.	Regional actions	19
8.2.	National actions	19
8.2.1.	ANR AutoCHEM	19
8.2.2.	ANR Arpège ASOPT: Analyse Statique et OPTimisation	19
8.2.3.	Action INRIA d'envergure nationale "Synchronics"	19
8.2.4.	Collaborations inside Inria	20
8.2.5.	Cooperations with other laboratories	20
8.3.	European actions	20
8.3.1.	Artist II European IST network of Excellence	20
8.3.2.	ArtistDesign European IST network of Excellence	21
8.3.3.	Combest European IST STREP	21
8.3.4.	AOSD European IST network of Excellence	21
8.3.5.	Interlink Action	21
8.4.	International actions	21
9.	Dissemination	21
9.1.	Scientific community	21
9.2.	Teaching	22
10.	Bibliography	22

1. Team

Research Scientist

Alain Girault [Team Leader, DR INRIA, HdR]
Pascal Fradet [CR INRIA, HdR]
Gregor Goessler [CR INRIA]
Bertrand Jeannot [CR INRIA]
Daniel Le Métayer [DR INRIA; since 06/2008 leader of the LICIT team, HdR]
Eric Rutten [CR INRIA, HdR]

External Collaborator

Gwenaël Delaval [MENRT grant]
Emil Dumitrescu [INSA Lyon]

PhD Student

Simplice Djoko Djoko [INRIA grant, AOSD NoE, until 11/2008]
Mouaiad Alras [Region Rhône-Alpes grant, ISLE cluster]
Gérald Vaisman [INRIA grant, DCN]
Nafees Qamar [INRIA Cordis, since 01/2008]
Marnes Hoff [INRIA grant, since 04/2008]
Soufyane Aboubekr [CORDIS, since 10/2008]
Lies Lakhdar-Chaouch [INRIA grant, OpenTLM]
Bhasker Vanamali [INRIA grant, OpenTLM, since 04/2008]

Post-Doctoral Fellow

Xavier Briand [INRIA]
Jean-Baptiste Raclet [COMBEST, since 02/2008]
Na Xu [COMBEST, since 17/11/2008]

Administrative Assistant

Helen Pouchot [Secretary (SAR) INRIA, until 06/2008]
Diane Courtiol [Secretary INRIA, since 06/2008]

Other

Nabil Kaabi [ENSI Tunis, 04-07/2008]

2. Overall Objectives

2.1. Overall Objectives

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 LUSTRE¹ and ESTEREL² languages) and fault tolerance (in relation with the SYNDEX³ 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 Mode Automata and SIGNAL⁴ 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:
 - ARTISTDESIGN⁵, about embedded real-time systems,
 - AOSD-Europe⁶, about formal methods for Aspect-Oriented Programming,
- a European STREP: COMBEST on component-based design;
- two ANR projects: ASOPT (on static analysis) and AUTOCHEM (on chemical programming);
- the OPENTLM project of the MINALOGIC Pole of Competitiveness, dedicated to the design flow for next generation SoC and SystemC,
- an industrial contract with DCNS.

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 ARTISTDESIGN (Advanced Real-Time Systems)⁷.

⁴<http://www.irisa.fr/espresso>

⁵<http://www.artist-embedded.org>

⁶<http://www.aosd-europe.net/>

⁷<http://www.artist-embedded.org/>

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 [36], [43]. During the past years, methodologies for the formal verification [74], [45], control synthesis [76] and compilation, and extensions to timed and hybrid systems [71], [37] 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 [65], which can be encoded in a synchronous model [39].

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 [40] 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].

The synchronous approach⁸ [63], [64] to reactive systems design gave birth to complete programming environments, with languages like ARGOS, LUSTRE⁹, ESTEREL¹⁰, SIGNAL/ POLYCHRONY¹¹, SYNDEX¹², LUCID SYNCHRONE¹³ or Mode Automata. 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.

⁸<http://www.synalp.org>

⁹<http://www-verimag.imag.fr/SYNCHRONE>

¹⁰<http://www.inria.fr/recherche/equipes/aoste.en.html>

¹¹<http://www.irisa.fr/espresso/Polychrony>

¹²<http://www-rocq.inria.fr/syindex>

¹³<http://www.lri.fr/~pouzet/lucid-synchrone/>

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. Application domains where the actors are ready to employ specialists in formal methods or advanced control theory are still uncommon. Even then, design methods based on the systematic application of these theoretical results are not ripe. In fields like industrial control, where the use of PLC (Programmable Logic Controller [41]) is dominant, this question can be decisive.

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 multidisciplinary 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 [73] and Mode Automata [72] 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/ dSPACE¹⁴) 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.

Regarding the synchronous approach, commercial tools are available: SCADE (based on LUSTRE), ESTEREL STUDIO¹⁵, SILDEX (based on SIGNAL), specialized environments like CELLCONTROL for industrial automation (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 DCNS on multi-criteria real-time scheduling issues for action planning of their defense systems.

5. Software

5.1. NBac

Participant: Bertrand Jeannot.

NBAC (Numerical and Boolean Automaton Checker)¹⁶ 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 two 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.

¹⁴<http://www.dspaceinc.com>

¹⁵<http://www.esterel-technologies.com>

¹⁶<http://pop-art.inrialpes.fr/people/bjeannot/nbac/>

It has been used for verification and debugging of LUSTRE programs [67] [52]. It is connected to the LUSTRE toolset¹⁷. 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 STG¹⁸ [46] [68] for selecting test cases.

5.2. Prometheus

Participant: Gregor Goessler.

The BIP component model (Behavior, Interaction model, Priority) [59][8] 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.

5.3. Implementations of synchronous programs

Participant: Alain Girault.

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 web¹⁹. 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 SYNDEX²⁰. 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 [54], 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) [55].

5.4. APRON libray

Participant: Bertrand Jeannet.

¹⁷<http://www-verimag.imag.fr/SYNCHRONE/index.php?page=tools>

¹⁸<http://www.irisa.fr/prive/plochette/stg-doc/stg-web.html>

¹⁹<http://pop-art.inrialpes.fr/people/girault/Ocrep/>

²⁰<http://www-rocq.inria.fr/syindex>

The APRON library²¹ 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.

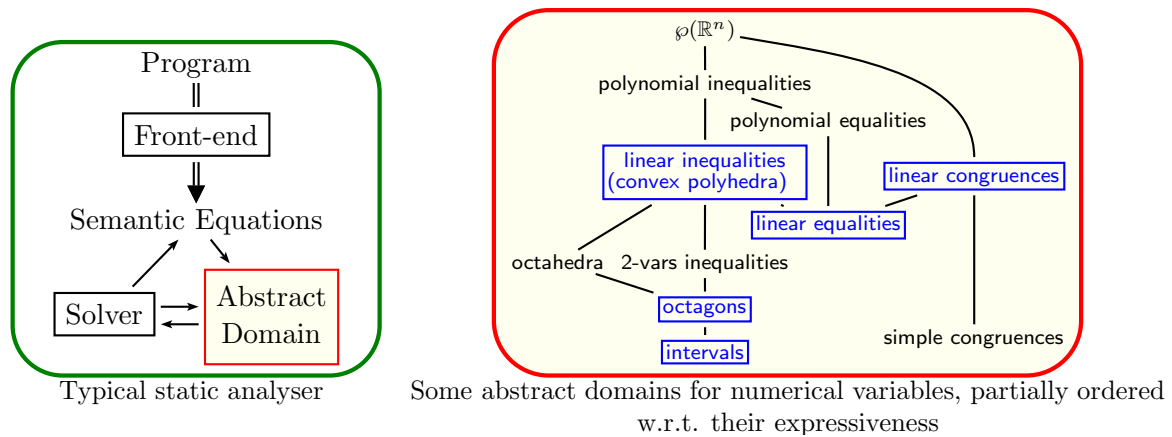


Figure 1. Typical static analyser and examples of abstract domains

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.

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 developer interface, using domain-independent datatypes, and exploiting common services. Independent libraries like PPL [38] 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.

²¹<http://apron.cri.ensmp.fr/library/>

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.enscm.fr/>.

Its development 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)

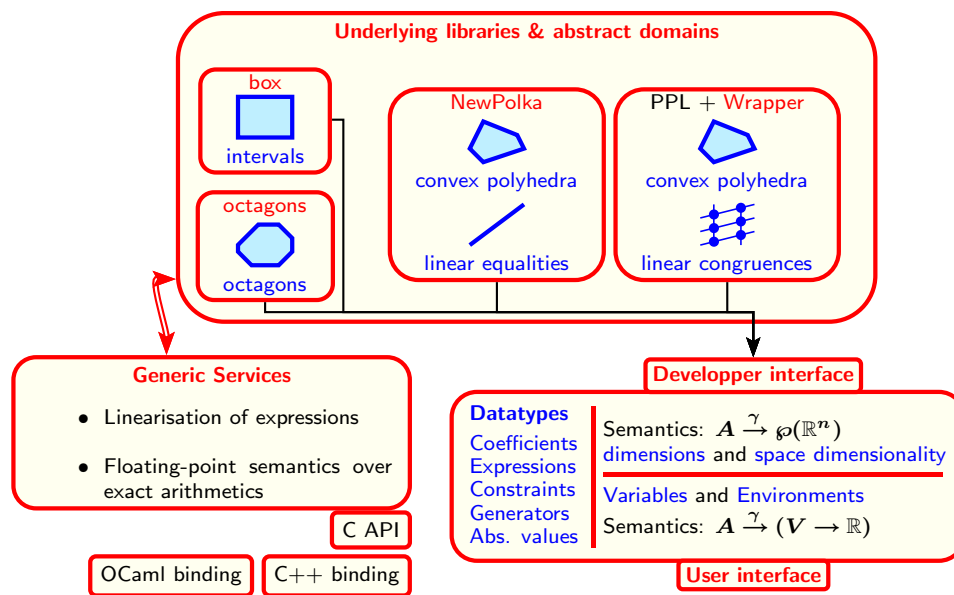


Figure 2. Organisation of the APRON library

5.5. Prototypes

5.5.1. Automatic Controller Generation

Participants: Gwenaël Delaval, Emil Dumitrescu, Alain Girault, Eric 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 SIGALI²² and of Mode Automata²³.

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).

5.5.2. Rapture

Keywords: Markov Decision Processes, Probabilistic verification.

Participant: Bertrand Jeannot.

²²<http://www.irisa.fr/vertecs/Logiciels/sigali.html>

²³<http://www-verimag.imag.fr>

RAPTURE [66] [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 the designer 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: Bertrand Jeannot.

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

Fixpoint ²⁴: 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 [60].

Interproc ²⁵: 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-interface²⁶. It has been cited in at least 3 published research papers in 2008.

ConcurInterproc extends Interproc with concurrency, for the analysis of multithreaded programs interacting via shared global variables. It is also deployed through a web-interface: ²⁷.

6. New Results

6.1. Dependable distributed real-time embedded systems

Participants: Alain Girault [contact person], Gérald Vaisman.

6.1.1. Static multiprocessor scheduling with tradeoff between performance and reliability

We have completed two works on this topic, both addressing the problem of bicriteria (length, reliability) scheduling. The first one defines a novel framework, where the reliability criterion is replaced by another criterion, called the global system failure rate (GSFR), which is independent from the length criterion [13]. We are now working on incorporating more reliability techniques into this framework, like the temporal redundancy, the k-out-of-n redundant systems, and the voting mechanisms. The second work restricts itself to a simpler case where the communication network is reliable. Thanks to this assumption we are able to propose a two-phase scheduling algorithm, where the first phase addresses only the reliability criterion by adequately replicating the tasks onto the processors (hence computing only the spatial allocation), while the second phase addresses only the length criterion by computing a temporal allocation that minimizes the schedule makespan. Thanks to this decoupling into two independent phases, we are able to produce approximate sets of Pareto non-dominated solutions for a given instance. Simulation results show that our algorithm outperforms, in terms of makespan, existing reference scheduling methods for heterogeneous processors (HEFT), while providing a better reliability [14].

²⁴<http://bjeannot.gforge.inria.fr/fixpoint>

²⁵<http://bjeannot.gforge.inria.fr/interproc>

²⁶<http://pop-art.inrialpes.fr/interproc/interprocweb.cgi>

²⁷<http://pop-art.inrialpes.fr/interproc/concurinterprocweb.cgi>

Unlike most work found in the literature, all our contributions are truly bicriteria in the sense that the user can gain several orders of magnitude on the reliability of his schedule, thanks to the active replication of tasks onto processors. In contrast, most of the other algorithms do not replicate the tasks, and hence have a very limited impact on the reliability.

6.1.2. *Static multiprocessor scheduling of tasks with resource constraints*

We continue our work on multi-criteria scheduling of tasks onto processors, where the main criteria are the reliability of the schedule, its makespan, and the number of used processors. In addition, assigning a task to a processor consumes a fixed number of limited resources available on this processor: we also want to minimize the total number of consumed resources. Finally, our scheduling problem is dynamic in the sense that some new tasks are created after each interval of one second while some old tasks are destroyed: as a consequence, a new schedule must be re-computed each second, and if possible there must be some “stability” between two successive schedules (meaning that tasks are common between two successive intervals should be assigned to the same respective processor in the two schedules). The solution that we explore is based on a branch and bound algorithm, 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.

6.2. Automatic distribution of synchronous programs

Participants: Mouaiad Alras, Gwenaël Delaval, Alain Girault [contact person].

6.2.1. *Modular distribution and application to discrete controller synthesis*

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) [10].

In order to address this problem, we have extended a synchronous data flow 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 [23].

First, a distributed semantics has been proposed in order to formalize the distributed execution of a program. Then, a type and effects system, 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.

Furthermore, we have shown the application of the automatic distribution of synchronous reactive programs to the specific problem of discrete controller synthesis of complex reactive systems. Discrete controller synthesis is a formal method used to ensure properties on a flexible system which does not a priori verify them. However, this method is efficient only on Boolean programs. More complex embedded systems, comprising complex data types and structures, cannot be addressed without abstraction means. We show how such abstractions can be obtained automatically using a type-directed projection operation. This operation allows then the safe recombination of the result of the synthesis with the original abstracted system, preserving the ensured properties [16].

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

Model-based design (MBD) involves designing a model of a control system, simulating and debugging it with dedicated tools, and finally generating automatically code corresponding to this model. In the domain of embedded systems, it offers the huge advantage of avoiding the time-consuming and error-prone final coding phase. The main issue raised by MBD is the faithfulness of the generated code with respect to the initial model, the latter being defined by the simulation semantics. To bridge the gap between the high-level model and the low-level implementation, we use the synchronous programming language Lustre as an intermediary formal model [62]. Concretely, starting from a high-level model specified in the de-facto standard Simulink, we first generate Lustre code along with some necessary structured “glue code”, and then we generate embedded real-time code for the Xenomai RTOS²⁸. Thanks to Lustre’s clean mathematical semantics, we are able to guarantee the faithfulness of the generated multi-tasked real-time code. 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: Soufyane Aboubekr, Gwenaël Delaval, Emil Dumitrescu, Alain Girault, Bertrand Jeannet, Nabil Kaabi, Eric Rutten.

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* [75], and more particularly following a methodology [1] adapted to the synchronous approach [73].

6.3.1. Distributed Controller Synthesis using Static Analysis of FIFO channels

We address the controller synthesis for distributed systems modeled as a set of sequential machines communicating via unbounded FIFO channels

The static analysis of stacks and FIFO queues was the topic of the PhD of Tristan Le Gall, defended in June 2008. We proposed in [9] a new abstract domain for languages on infinite alphabets, which acts as a functor taking an abstract domain for a concrete alphabet, and lifts it to an abstract domain for words on this alphabet.

We studied this year the application of this technique to the controller synthesis of a set of sequential machines communicating via unbounded FIFO channels, for which we consider simple state-avoidance properties. It is well-known that there exists no optimal (most permissive) controller in such a context, so our ambition is to propose a technique for computing “permissive-enough” controller. Our approach is based on the computation of global controller, which is then projected on local sites so as to obtain a controller per site in the controlled distributed system. We exploit the abstract domain mentioned above in the fixpoint computations involved in the computation of a correct controller, and we take into account the following partial observation constraints: the global controller cannot observe the contents of the FIFO channels to take its decisions (they model communication links), and the projected local controller has a knowledge only on their local state.

A technical report on this approach is being written, and we plan to submit a paper at the beginning of 2009. This work is conducted in collaboration with H. Marchand and T. Le Gall (VERTECS team from INRIA Rennes).

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

We have proposed a simple programming language, called NEMO [4], specific to the domain of multi-task real-time control systems. We obtain automatically an application-specific task handler that correctly manages the constraints, through a compilation-like process including a phase of discrete controller synthesis.

²⁸<http://www.xenomai.org>

We have extended this work in the directions of:

- the definition of a more general modeling and programming language,
- the exploitation of modularity in synthesis, encapsulated in the compilation, in order to make larger models amenable to our techniques.

This work is conducted in collaboration with H. Marchand and G. Delaval (VERTECS team from INRIA Rennes).

6.3.3. Automating the addition of fault tolerance with discrete controller synthesis

We have defined a new framework for the design of fault tolerant embedded systems, based on Discrete controller synthesis (DCS), a formal approach based on the same state-space exploration algorithms as model-checking [75]. We start with a fault intolerant program, modeled as the synchronous parallel composition of finite labeled transition systems; we specify formally a fault hypothesis; we state some fault tolerance requirements; and we use DCS to obtain automatically a program, having the same behavior as the initial fault intolerant one in the absence of faults, and satisfying the fault tolerance requirements under the fault hypothesis. Our original contribution resides in the demonstration that DCS can be elegantly used to design fault tolerant systems, with guarantees on key properties of the obtained system, such as the fault tolerance level, the satisfaction of quantitative constraints, and so on. We show with numerous examples taken from case studies (some of them previously published [56], [51], [57], [50]) that our method can address different kinds of failures (crash, value, or Byzantine) affecting different kinds of hardware components (processors, communication links, actuators, or sensors). Besides, we show that our method also offers an optimality criterion very useful to synthesize fault tolerant systems compliant to the constraints of embedded systems, like power consumption. In summary, the strong advantages of our framework for fault tolerance are:

- It is **automatic**, because DCS produces automatically a fault tolerant system from an initial fault intolerant one.
- The **separation of concerns**, because the fault intolerant system can be designed independently from the fault tolerance requirements.
- The **flexibility**, because, once the system is entirely modeled, it is easy to try several fault hypotheses, several environment models, several fault tolerance goals, several degraded modes, and so on.
- The **safety**, because, in case of positive result obtained by DCS, the specified fault tolerance properties are guaranteed by construction on the controlled system.
- The **optimality** when optimal synthesis is used, modulo the potential numerical equalities (hence a non-strict optimality).

We are extending this work in the direction of optimal synthesis considering weights cumulating along bounded-length paths, and its application to the control of sequences of reconfigurations. We are adapting models in order to take into account the additive costs of e.g., execution time or power consumption, and adapting synthesis algorithms in order to support the association of costs with transitions, and the handling of these new weight functions in the optimal synthesis. 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 cumulated consumption of the resulting dynamically reconfiguring fault tolerant system.

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

6.3.4. Model-based control of adaptive 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 the 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 have defined a framework for the model-based control of adaptive

systems, especially, considering their discrete aspects [34], [21]. We base it on our previous work, that gave contributions to different separate issues related to this topic; we are working on a generalisation towards adaptive systems, with applications in embedded middleware for autonomic systems, and reconfigurable architectures.

An experiment has started on the modeling of task management mechanism of the ORCCAD environment developed in the NECS team (D. Simon and R. Pissard-Gibollet), and the application of discrete controller synthesis.

This work is conducted in contact with the SARDES (at the middleware level) and NECS (at the task management level) teams of INRIA in Grenoble.

6.4. Static Analysis and Abstract Interpretation

Participants: Xavier Briand, Bertrand Jeannot [contact person].

6.4.1. *Combining control and data abstraction in the verification of hybrid systems*

We have studied the verification of hybrid systems built as the composition of a discrete software controller interacting with a physical environment exhibiting a continuous behavior. Our goal is to attack the problem of the combinatorial explosion of discrete states that may happen if a complex software controller is considered. We propose to extend an existing abstract interpretation technique, namely dynamic partitioning, to hybrid systems. Dynamic partitioning, which shares some common principles with predicate abstraction, allows us to finely tune the tradeoff between precision and efficiency in the analysis.

We have extended the NBAC tool (Section 5.1) according to these principle, and showed the efficiency of the approach by a case study that combines a non trivial controller specified in the synchronous dataflow programming language LUSTRE with its physical environment.

6.4.2. *A Relational Approach to Interprocedural Shape Analysis*

This work addresses the verification of properties of imperative programs with recursive procedure calls, heap-allocated storage, and destructive updating of pointer-valued fields, i.e., interprocedural shape analysis. It presents a way to apply some previously known approaches to interprocedural dataflow analysis — which in past work have been applied only to a much less rich setting — so that they can be applied to programs that use heap-allocated storage and perform destructive updating.

This year we have extended the work in the conference paper [69] and we have submitted a journal paper to TOPLAS, which has been accepted in October [15].

This work has been done in collaboration with T. Reps (Univ. of Madison-Wisconsin), M. Sagiv (Univ. of Tel-Aviv) and A. Loginov (GrammarTech).

6.4.3. *Relational interprocedural analysis of concurrent programs*

We have studied the extension of the relational approach to interprocedural analysis of sequential programs to concurrent programs, composed of a fixed number of threads.

In the relational approach, a sequential program is analyzed by computing summaries of procedures, and by propagating reachability information using these summaries. We propose an extension to concurrent programs, which is technically based on an instrumentation of the standard operational semantics, followed by an abstraction of tuple of call-stacks into sets. This approach allows us to extend relational interprocedural analysis to concurrent programs. We have implemented it for programs with scalar variables, in the CONCURINTERPROC online analyzer (see Section 5.5.3).

We have experimented several classical synchronisation protocols in order to investigate the precision of our technique, but also to analyze the approximations it performs.

We have published a research report [33] on these results.

6.5. Component-Based Construction

Participants: Alain Girault, Gregor Goessler [contact person], Jean-Baptiste Raclet, Na Xu.

6.5.1. Specification enforcing refinement for convertibility verification

Protocol conversion deals with the automatic synthesis of an additional component or glue logic, often referred to as an *adaptor* or an *interface*, to bridge mismatches between interacting components, often referred to as *protocols*. A formal solution, called convertibility verification, has been recently proposed, which produces such a glue logic, termed as a *converter*, so that the parallel composition of the protocols and the converter also satisfies some desired specification. A converter is responsible for bridging different kinds of mismatches such as *control*, *data*, and *clock* mismatches. Mismatches are usually removed by the converter (similar to controllers in supervisory control of discrete event systems) by *disabling undesirable paths* in the protocol composition.

We have formulated a generalization of this convertibility verification problem, by using a new refinement relation called *Specification Enforcing Refinement (SER)* between a protocol composition and a desired specification. The existence of such a refinement is shown to be a necessary and sufficient condition for the existence of suitable a converter. We have also proposed an approach to automatically synthesize a converter if a SER refinement relation exists. The proposed converter is capable of the usual disabling actions to remove undesirable paths in the protocol composition. In addition, the converter can perform *forcing actions* when disabling alone fails to find a converter to satisfy the desired specification. Forcing allows the generation of control inputs in one protocol that are not provided by the other protocol. Forcing induces *state-based hiding*, an operation not achievable using discrete event systems control theory.

6.5.2. Compositional strategy mapping

In the context of our work on compositionality and reconfigurability, we are studying the issue of implementing a component system on a lower-level platform. With the increasing complexity of embedded systems, coupled with the need for faster time-to-market and high confidence in the reliability of the product, design methods that ensure correctness by construction are, when available, the solution of choice. When dealing with reactive systems, which interact with their environment, the behavior of the system to be designed has to be considered in terms of *strategies*: can some desired behavior be enforced in spite of the — potentially non cooperative — environment?

Computing a strategy satisfying some property is expensive, and although modular and compositional controller synthesis have been studied for some decades, this remains a hard problem. In particular, progress properties are notoriously more difficult to tackle compositionally than safety properties.

We are interested in a design flow supporting the refinement of strategies, rather than in controller synthesis performed on some given level of abstraction. We consider a platform-based design process consisting of successive mapping steps [70]. The goal of each step is to constructively map a strategy constructed so far onto a lower-level platform. The mapping is performed component-wise, using an abstraction of the environment of each component. We have developed compositionality results ensuring that the refinement carries over to the global strategy [20].

As the analysis of reachability properties is often crucial to understand the behavior of complex systems, we consider *reach strategies* as a special case of strategies. Reach strategies generalize the notion of acyclic paths in closed systems, ensuring reachability under control of a set of states.

6.5.3. Contract-based design

We are investigating a contract-based design flow where contracts are tuples (assumption, guarantee) of modal automata. The consideration of modalities allows us to specify the level of certainty and hence quality of service constraints. The composition operation on contracts, based on residuation, has been defined so as to satisfy the properties of correctness, completeness (under some reasonable sufficient condition), associativity, and stepwise refinement. We are currently studying a design flow supporting multiple aspects and shared refinement, based on this composition operation, and its application to the BIP component framework.

6.6. Aspect-oriented programming

Participants: Simplice Djoko Djoko, Pascal Fradet [contact person], Alain Girault, Nafees Qamar.

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 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 [17]. It is sufficient to check that an aspect belongs to a specific category to know a large class of properties that will be preserved by weaving. Our categories of aspects comprise, among others, 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 the executions remain in the set of reachable states of the base program (*e.g.*, optimization or fault-tolerance aspects). These categories are defined formally based on a language independent abstract semantic framework. The classes of properties are defined as subsets of LTL for deterministic programs and 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.

In a second step, we have designed for each aspect category a specialized aspect language which ensures that any aspect written in that language belongs to the corresponding category [18]. These languages preserve the corresponding classes of properties by construction. The aspect languages share the same expressive pointcut language and are designed *w.r.t.* a common imperative base language.

This work is the central topic of Simplice Djoko Djoko's PhD thesis which should be defended in the beginning of 2009. It was conducted within the Formal Methods Lab of the network of excellence AOSD-Europe (see Section 8.3.4) 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 [30]. Our technique allows us to keep resource management and availability issues separate from the rest of the system.

We have proposed a domain-specific aspect language aimed at preventing denial of service caused by resource management (*e.g.*, starvation, deadlocks, etc.). The 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. 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 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.

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

Here, our objective is to design a domain-specific language for specifying fault tolerance aspects 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. We are currently working on the definition of an aspect language allowing the designer to specify such choices. This work is conducted in collaboration with T. Ayav from Izmir University.

6.7. Other results

6.7.1. *Chemical programming*

Participants: Pascal Fradet [contact person], Marnes Hoff.

The chemical reaction metaphor describes computation in terms of a chemical solution in which molecules (representing data) interact freely according to reaction rules (representing the program). 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 as well as autonomicity (e.g. self-healing, self-protection, self-optimization, etc.) are naturally expressed as reaction rules.

In the context of the InterLink coordination action (see Section 8.3.5), we have written a review of recent advances concerning the chemical reaction model [25]. This work presents several generalizations of the basic model (higher-order reactions, hybrid and infinite multisets) and current research directions. We have also described classical coordination mechanisms and parallel programming models (Linda, Petri Nets, Kahn Networks) in the same chemical setting [24]. All these examples put forward the simplicity and expressivity of the chemical paradigm.

This work was conducted in collaboration with Jean-Pierre Banâtre and Yann Radenac from the PARIS project team at IRISA.

A drawback of chemical languages is that their very high-level nature usually leads to very inefficient programs. We are currently looking at approaches to refine chemical programs to more efficient lower-level programs. Several research directions are promising: richer types and data structures, domain-specific languages or aspects describing separately implementation issues (such as the evaluation strategy or data representation).

This line of research is part of Marnes Hoff's PhD thesis and takes place within the AUTOCHEM project (see Section 8.2.1).

6.7.2. *Component-based modeling and reachability analysis of genetic networks*

Participant: Gregor Goessler.

Genetic regulatory networks usually encompass a multitude of complex, interacting feedback loops. Being able to model and analyze their behavior is crucial for understanding the interactions between the proteins, and their functions. Genetic regulatory networks have been modeled as discrete transition systems by many approaches, benefiting from a large number of formal verification algorithms available for the analysis of discrete transition systems. However, most of these approaches face the problem of state space explosion, as even models of modest size (from a biological point of view) usually lead to large transition systems, due to a combinatorial blow-up of the number of states. This problem has been addressed with the component-based approach of [58] — based on the mathematically well-founded formalism of qualitative simulation [77] — 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, and provided both correctness and completeness results [32].

Using the same approach, we are studying, in cooperation with H. de Jong (IBIS project team) and G. Batt (CONSTRAINTES project team), the definition of a symbolic representation of the network behavior as a compact exchange format between the Genetic Network Analyzer (GNA) developed in the IBIS project team, and the model checker CADP developed by VASY.

6.7.3. Control for data-parallel systems

Participant: Eric Rutten.

Data intensive computing is increasingly getting high importance in a wide range of scientific and engineering domains, like multi-media portable devices. 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 GASPARD2²⁹ 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 [19], [12], in order to bridge the gap between GASPARD2 and analysis and verification tools of the synchronous technology so that formal validation is favored [22]. The automation of the transformations is implemented within an MDE framework [26]. We extended GASPARD2, by adding reactive control features based on finite state machines, and are integrating this extension in the synchronous model [31].

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

7. Contracts and Grants with Industry

7.1. Pôle de compétitivité Minalogic: OpenTLM

In the context of the pôle de compétitivité Minalogic, we participate in the four-year project OpenTLM on analysis of systems-on-chip modeled at the transaction level in SystemC [61]. We intend to develop methods for abstraction, and interprocedural and compositional analysis of SystemC models. Preliminary results have been obtained, see Section 6.4.3. Two PhD students have been hired on these topics, resp. in April and May 2008.

7.2. DCNS

With the INRIA project team MOAIS and the ProBayes start-up, we have signed a contract with DCNS. DCNS 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 (see Section 6.1.2).

²⁹<http://www.lifl.fr/west/gaspard>

7.3. Pôle de compétitivité Minalogic: MIND

We participate in the MIND project supported by the MINALOGIC competitiveness cluster; its general purpose is the consolidation of the Fractal approach to embedded middleware, proposed by the SARDES team. Within the project, we are contributing to the model-based control of adaptative systems, for the management of fault tolerance or quality or service, applying discrete controller synthesis (see Section 6.3.4).

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 6.2.2).

8.2. National actions

8.2.1. ANR AutoCHEM

Participants: Pascal Fradet [contact person], Marnes Hoff.

The AUTOCHEM project aims at investigating and exploring the use of chemical languages (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.

8.2.2. ANR Arpège ASOPT: Analyse Statique et OPTimisation

Participants: Bertrand Jeannet, Lies Lakhdar-Chaouch.

The ASOPT (Analyse Statique et OPTimisation) project [end of 2008-2011] brings together static analysis (INRIA-POP ART, VERIMAG, CEA LMeASI), optimisation, and control/game theory experts (CEA LMeASI, INRIA-MAXPLUS) around some program verification problems.

Many abstract interpretations attempt to find "good" geometric shapes verifying certain constraints; this not only applies to purely numerical abstractions (for numerical program variables), but also to abstractions of data structures (arrays and more complex shapes). This problem can often be addressed by optimisation techniques, opening the possibility of exploiting advanced techniques from mathematical programming.

The purpose of ASOPT is to develop new abstract domains and new resolution techniques for embedded control programs, and in the longer run, for numerical simulation programs.

POP ART is the coordinator of the project.

8.2.3. Action INRIA d'envergure nationale "Synchronics"

Participants: Alain Girault, Bertrand Jeannet.

The SYNCHRONICS (Language Platform for Embedded System Design) project [beginning of 2008-2011] gathers 9 permanent researchers on the topic of embedded systems design: B. Caillaud (IRISA), A. Cohen, L. Mandel, & M. Pouzet (INRIA-Saclay), A. Girault & B. Jeannet (INRIA-Grenoble – Rhône-Alpes), E. Jahier & P. Raymond (VERIMAG).

SYNCHRONICS capitalizes on recent extensions of data-flow synchronous languages, as well as relaxed forms of synchronous composition or compilation techniques for various platform, to address two main challenges with a language-centered approach: (i) the co-simulation of mixed discrete-continuous specifications, and more generally the co-simulation of programs and properties (either discrete or continuous); (ii) the ability, inside the programming model, to account for the architecture constraints (execution time, memory footprint, energy, power, reliability, etc.).

8.2.4. 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 and T. Priol (PARIS, IRISA/INRIA-Rennes) and with R. Douence (OBASCO, Ecole des Mines de Nantes).
- A. Girault cooperates with the MOAIS project (CR Grenoble – 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 D. Le Métayer (LICIT action, CR Rhône-Alpes), H. de Jong (IBIS project, CR Rhône-Alpes), and G. Batt (CONTRAINTES project, CR Rocquencourt).
- B. Jeannet cooperates with T. Le Gall (VERTECS, IRISA/INRIA-Rennes) on the analysis of communicating systems, and with C. Constant, T. Jérón and F. Ployette (VERTECS, IRISA/INRIA-Rennes) on test generation.
- E. Rutten is working with VERTECS at IRISA/INRIA-Rennes on applications of discrete controller synthesis, and in particular on the tool SIGALI. He is working with the DART project at CR Futurs in Lille, on the synchronous modelling of massively parallel application, and the introduction of control and mode automata in the GASPARD framework. He is working on model-based control of adaptive systems with SARDES (at the middleware level) and with NECS (at the task management level).

8.2.5. Cooperations with other laboratories

- P. Fradet cooperates with J.-L. Giavitto (CNRS/Université d'Evry) and S. Hong Tuan Ha (CEA Saclay).
- A. Girault cooperates with X. Nicollin (VERIMAG), M. Pouzet (LRI, University of Paris VI), D. Trystram and É. Saule from (ID-IMAG), C. Dima (Université of Paris XII), and H. Kalla (University of Batna, Algeria).
- P. Fradet and A. Girault collaborate with T. Ayav (University of Izmir, Turkey).
- G. Goessler cooperates with J. Sifakis and S. Bensalem (VERIMAG) and M. Majster-Cederbaum (University of Mannheim, Germany).
- B. Jeannet cooperates with N. Halbwegs and M. Péron (VERIMAG) on static analysis and abstract interpretation.
- E. Rutten collaborates with H. Alla (GIPSA, Grenoble).

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 design³⁰. 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). G. Goessler is the administrator of ARTIST II for INRIA. ARTIST II ended in September 2008.

³⁰<http://www.artist-embedded.org/FP6>

8.3.2. *ArtistDesign European IST network of Excellence*

ARTISTDESIGN is a European Network of Excellence on embedded system design, successor of ARTIST II in FP7. The objective for ARTISTDESIGN is to build on existing structures and links forged in ARTIST II, to become a virtual Center of Excellence in Embedded Systems Design. This will be mainly achieved through tight integration between the central players of the European research community. The long-term vision for embedded systems in Europe, established in ARTIST II, will advance the emergence of Embedded Systems as a mature discipline. G. Goessler is the administrator of ARTISTDESIGN for INRIA.

8.3.3. *Combest European IST STREP*

COMBEST is a European STREP on formal component-based design of complex embedded systems³¹. Its goal is to develop a design theory for embedded systems, covering heterogeneity, interface specifications, composability, compositionality, and refinement for functional and extra-functional properties.

8.3.4. *AOSD European IST network of Excellence*

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

8.3.5. *Interlink Action*

InterLink is a coordination action funded by the European Commission. It aims at advancing Europe's knowledge in a number of critical Information and Communication Science and Technologies areas. P. Fradet participates in the working group on software-intensive systems and novel computing paradigms.

8.4. International actions

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 H. Alla). The work is related to Model-based control of adaptative systems (see section 6.3.4).

9. Dissemination

9.1. Scientific community

- P. Fradet participates in the program committees of RV'09 (*International Workshop on Runtime Verification*) and FOAL'09 (*Foundations of Aspect-Oriented Languages Workshop*).
- A. Girault co-chaired the programme committee of SLA++P'08, the workshop on Model-driven High-level Programming of Embedded Systems, co-located with ETAPS'08, and served in the programme committee of the DATE'08 Conference³². He was PhD assessor for the jury of Julien Boucaron (University of Nice).
- A. Girault and E. Rutten co-chaired SLA++P'08, the workshop on Model-driven High-level Programming of Embedded Systems, co-located with ETAPS'08 [27]. They are co-editors, with F. Maraninchi, M. Mendler, and M. Pouzet, of the special issue of the Journal on Embedded Systems publishing selected papers from SLA++P'07 and SLA++P'08" [28].

³¹<http://www.combest.eu>

³²DATE'08: <http://www.date-conference.com>

- G. Goessler is a guest editor of a special issue of EURASIP Journal on Embedded Systems on “Formal Techniques for Embedded Systems Design and Validation”. He was also member of the PhD committees of J.-B. Raclet (U. Rennes/IRISA) and R. Sinha (U. Auckland).
- E. Rutten is co-editor, with J.-M. Muller and E. Niel, of the special issue of the JUCS Journal on “Interactions of discrete-event automation and computer sciences” [29]. He participated in the program committee of the Workshop on Adaptive and Reconfigurable Embedded Systems APRES’08³³, co-located with IEEE RTAS’08 at CPSWEEK. He is in charge of coordination between INRIA and CEA-DRT on Systems-on-Chip. He was member of the PhD committee of S. Taha (U. Lille 1/CEA-LIST). He is a member of the *Commission de spécialistes* (recruiting committee) at the Universities of Brest (UBO) and Lille-1.

9.2. Teaching

9.2.1. Advising

PhDs:

- Gwenaël Delaval, co-advised by Alain Girault (with M. Pouzet, LRI Orsay), since 09/2004. PhD in computer science, INPG, defended on July 1st, 2008.
- Tristan Le Gall, co-advised by Bertrand Jeannot (with T. Jéron, VERTECS IRISA) since 09/2004. PhD in computer science, University of Rennes I, defended on July 2, 2008.
- Camille Constant, co-advised by Bertrand Jeannot (with T. Jéron, VERTECS IRISA) since 9/2006. PhD in computer science, University of Rennes I, defended on November 24, 2008.
- Simplice Djoko Djoko, co-advised by P. Fradet (with R. Douence, OBASCO, Ecole des Mines de Nantes), since 09/2005, PhD in computer science, University of Nantes.
- Nafees Qamar, advised by Pascal Fradet, since 01/2008, PhD in computer science, UJF, Grenoble.
- Marnes Hoff, co-advised by P. Fradet (with J.-L. Giavitto, Université d’Evry), since 04/2008, PhD in computer science, INPG.
- Mouaiad Alras, co-advised by Alain Girault (with P. Raymond, VERIMAG Grenoble), since 10/2006, PhD in computer science, UJF, Grenoble.
- Gérald Vaisman, co-advised by Alain Girault (with P.-F. Dutot, MOAIS CR Rhône-Alpes), since 10/2006, PhD in computer science, INPG.
- Huafeng Yu, co-advised by E. Rutten (with J.-L. Dekeyser, LIFL/INRIA Futurs Lille), from 10/2005 to 11/2008. PhD in computer science, University of Lille 1.
- Soufyane Aboubekr, advised by E. Rutten, since 10/2008, PhD in computer science, INPG.

Masters:

- Ezequiel Geremia, 04-10/2008, Master’s thesis, on the modular analysis of synchronous programs.
- Nabil Kaabi, 04-07/2008, internship, Engineer in computer science, ENSI, Tunis.

10. Bibliography

Major publications by the team in recent years

- [1] K. ALTISEN, A. CLODIC, F. MARANINCHI, E. 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° 2618, Springer Verlag, April 2003, p. 174–188.

³³APRES’08: <http://www.artist-embedded.org/artist/APRES08.html>

- [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] T. AYAV, P. FRADET, A. GIRAULT. *Implementing fault-tolerance in real-time programs by automatic program transformations*, in "ACM Transactions on Embedded Computing Systems (TECS)", vol. 7, n^o 4, July 2008, p. 1-43.
- [4] G. DELAVAL, E. RUTTEN. *A domain-specific language for multi-task systems, applying discrete controller synthesis*, in "Journal on Embedded Systems (special issue on Synchronous Paradigm in Embedded Systems)", vol. 2007, n^o 84192, January 2007, 17, <http://www.hindawi.com/GetArticle.aspx?doi=10.1155/2007/84192>.
- [5] P. FRADET. *Approches langages pour la conception et la mise en œuvre de programmes*, Habilitation thesis, Université de Rennes 1, November 2000.
- [6] 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.
- [7] A. GIRAULT, H. KALLA. *A Novel Bicriteria Scheduling Heuristics Providing a Guaranteed Global System Failure Rate*, in "TDSC", Research report INRIA 6319, 2009, <http://hal.inria.fr/inria-00177117>.
- [8] G. GOESSLER, J. SIFAKIS. *Composition for Component-based Modeling*, in "Science of Computer Programming", vol. 55, n^o 1-3, 2005, p. 161-183.
- [9] 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>.

Year Publications

Doctoral Dissertations and Habilitation Theses

- [10] G. DELAVAL. *Répartition Modulaire de Programmes Synchrones*, Ph. D. Thesis, INPG, INRIA Grenoble Rhône-Alpes, projet Pop-Art, July 2008.

Articles in International Peer-Reviewed Journal

- [11] T. AYAV, P. FRADET, A. GIRAULT. *Implementing fault-tolerance in real-time programs by automatic program transformations*, in "ACM Transactions on Embedded Computing Systems (TECS)", vol. 7, n^o 4, July 2008, p. 1-43.
- [12] A. GAMATIÉ, E. RUTTEN, H. YU, P. BOULET, J.-L. DEKEYSER. *Synchronous Modeling and Analysis of Data Intensive Applications*, in "Journal on Embedded Systems", vol. 2008, n^o 561863, September 2008, 22, <http://www.hindawi.com/getarticle.aspx?doi=10.1155/2008/561863>.
- [13] A. GIRAULT, H. KALLA. *A Novel Bicriteria Scheduling Heuristics Providing a Guaranteed Global System Failure Rate*, in "IEEE Trans. Dependable Secure Comput.", Research report INRIA 6319., 2009, <http://hal.inria.fr/inria-00177117>.

- [14] A. GIRAULT, E. SAULE, D. TRYSTRAM. *Reliability Versus Performance for Critical Applications*, in "J. of Parallel and Distributed Computing", 2009.
- [15] B. JEANNET, A. LOGINOV, T. REPS, M. SAGIV. *A relational approach to interprocedural shape analysis*, in "ACM Trans. On Programming Languages and Systems (TOPLAS)", 2009.

International Peer-Reviewed Conference/Proceedings

- [16] G. DELAVAL, A. GIRAULT, M. POUZET. *A Type System for the Automatic Distribution of Higher-order Synchronous Dataflow Programs*, in "International Conference on Languages, Compilers, and Tools for Embedded Systems, LCTES'08, Tucson (AZ), USA", ACM, New-York, June 2008, p. 101–110, <ftp://ftp.inrialpes.fr/pub/bip/pub/girault/Publications/Lctes08/main.pdf>.
- [17] 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)", ACM, january 2008, p. 135-145.
- [18] S. DJOKO DJOKO, R. DOUENCE, P. FRADET. *Specialized Aspect Languages Preserving Classes of Properties*, in "Proc. of the Sixth IEEE International Conference on Software Engineering and Formal Methods (SEFM'08)", IEEE, november 2008, p. 227-236.
- [19] A. GAMATIÉ, E. RUTTEN, H. YU, P. BOULET, J.-L. DEKEYSER. *Modeling and Formal Validation of High-Performance Embedded Systems*, in "Proceedings of the 7th International Symposium on Parallel and Distributed Computing, ISPDC '08, Krakow, Poland July 1-5, 2008", 2008.
- [20] G. GOESSLER. *Compositional Strategy Mapping*, in "Proc. FSEN'09", F. ARBAB, M. SIRJANI (editors), LNCS, Springer, 2009.
- [21] E. RUTTEN. *Reactive control of adaptive embedded systems – a position paper*, in "7th Workshop on Adaptive and Reflective Middleware (ARM'08)", 2008.
- [22] H. YU, A. GAMATIÉ, E. RUTTEN, J.-L. DEKEYSER. *Safe Design of High-Performance Embedded Systems in a MDE Framework*, in "Proceedings of the First IEEE International workshop UML and Formal Methods, UML&FM'08", Special issue of the ISSE NASA journal Innovations in Systems and Software Engineering, 2008.

Workshops without Proceedings

- [23] G. DELAVAL. *Modular Distribution and Application to Discrete Controller Synthesis*, in "International Workshop on Model-driven High-level Programming of Embedded Systems, SLA++P'08, Budapest, Hungary", ENTCS, March 2008.

Scientific Books (or Scientific Book chapters)

- [24] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *Classical Coordination Mechanisms in the Chemical Model*, in "From Semantics to Computer Science: Essays in Honor of Gilles Kahn", Cambridge University Press, 2008.
- [25] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *The Chemical Reaction Model - Recent Developments and Prospects*, in "Software-Intensive Systems and New Computing Paradigms", Lecture Notes in Computer Science, vol. 5380, Springer, 2008, p. 209-234.

- [26] H. YU, A. GAMATIÉ, E. RUTTEN, J.-L. DEKEYSER. 13, in "Model Transformations from a Data Parallel Formalism towards Synchronous Languages", Lecture Notes Electrical Engineering Vol. 10, ISBN: 978-1-4020-8296-2, Springer Verlag, 2008.

Books or Proceedings Editing

- [27] A. GIRAULT, E. RUTTEN (editors). *International Workshop on Model-driven High-level Programming of Embedded Systems, SLA++P'08*, ENTCS, vol. To appear, March 2008.
- [28] F. MARANINCHI, M. MENDELER, M. POUZET, A. GIRAULT, E. RUTTEN (editors). *Model-driven High-level Programming of Embedded Systems. Selected papers from Sla++p'07 and Sla++p'08*, Special issue of the Journal on Embedded Systems, 2008, <http://www.hindawi.com>.
- [29] E. NIEL, E. RUTTEN, J. MULLER (editors). *Interactions of discrete-event automation and computer sciences*, Special issue of the Journal of Universal Computer Science, 2009, <http://www.jucs.org/jucs>.

Research Reports

- [30] P. FRADET, S. HONG TUAN HA. *Aspects of Availability - Enforcing timed properties to prevent denial of service*, (submitted to journal publication), Research Report, n^o 6754, INRIA, december 2008.
- [31] A. GAMATIÉ, E. RUTTEN, H. YU. *A Model for the Mixed-Design of Data-Intensive and Control-Oriented Embedded Systems*, Rapport de Recherche, n^o 6589, INRIA, July 2008, <http://hal.inria.fr/inria-00293909>.
- [32] G. GOESSLER. *Component-based Modeling and Reachability Analysis of Genetic Networks*, research report, n^o RR-6755, INRIA, 2008.
- [33] B. JEANNET. *Relational interprocedural analysis of concurrent programs*, Technical report, n^o 6671, INRIA, October 2008.
- [34] E. RUTTEN. *Reactive control of adaptive embedded systems*, Rapport de Recherche, n^o 6604, INRIA, July 2008, <http://hal.inria.fr/inria-00308660>.

Scientific Popularization

- [35] A. GIRAULT, H. KALLA. *Bicriteria Multi-Processor Static Scheduling*, October 2008, <http://ercim-news.ercim.org>, ERCIM News, 75:46–47.

References in notes

- [36] A. ARNOLD. *Systèmes de transitions finis et sémantique des processus communicants*, Masson, 1992.
- [37] 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.
- [38] R. BAGNARA, E. RICCI, E. ZAFFANELLA, P. M. HILL. *Possibly not Closed Convex Polyhedra and the Parma Polyhedra Library*, in "Static Analysis Symposium, SAS'02", LNCS, vol. 2477, 2002, <http://www.cs.unipr.it/~bagnara/Papers/Abstracts/SAS02>.

- [39] J.-R. BEAUVAIS, E. RUTTEN, T. GAUTIER, R. HOUEBINE, P. LE GUERNIC, Y.-M. 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.
- [40] R. BRYANT. *Graph-based algorithms for boolean function manipulation*, in "IEEE Transactions on Computers", vol. C-35, n^o 8, 1986, p. 677–692.
- [41] 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.
- [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, <http://doi.acm.org/10.1145/93542.93585>.
- [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] E. DUMITRESCU, A. GIRAULT, H. MARCHAND, E. RUTTEN. *Optimal Discrete Controller Synthesis for Modeling Fault-Tolerant Distributed Systems*, in "Workshop on Dependable Control of Discrete Systems, DCDS'07, Cachan, France", IFAC, New-York, June 2007, p. 23–28.
- [51] E. DUMITRESCU, A. GIRAULT, E. RUTTEN. *Validating Fault-Tolerant Behaviors of Synchronous System Specifications by Discrete Controller Synthesis*, in "Workshop on Discrete Event Systems, WODES'04, Reims, France", IFAC, New-York, September 2004.
- [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, 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.
- [55] 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.
- [56] A. GIRAULT, E. RUTTEN. *Discrete Controller Synthesis for Fault-Tolerant Distributed Systems*, in "International Workshop on Formal Methods for Industrial Critical Systems, FMICS'04, Linz, Austria", ENTCS, vol. 133, Elsevier Science, New-York, September 2004, p. 81–100.
- [57] A. GIRAULT, H. YU. *A Flexible Method to Tolerate Value Sensor Failures*, in "International Conference on Emerging Technologies and Factory Automation, ETFA'06, Prague, Czech Republic", IEEE, Los Alamitos, CA, September 2006, p. 86–93.
- [58] G. GOESSLER. *Compositional Reachability Analysis of Genetic Networks*, in "CMSB'06", C. PRIAMI (editor), LNBI, vol. 4210, Springer, 2006, p. 212-226.
- [59] 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.
- [60] D. GOPAN, T. REPS. *Guided Static Analysis*, in "Static Analysis Symposium, SAS'07", LNCS, vol. 4634, August 2007, http://dx.doi.org/10.1007/978-3-540-74061-2_22.
- [61] T. GRÖTKER, S. LIAO, G. MARTIN, S. SWAN. *System Design with SystemC*, Kluwer, 2002.
- [62] N. HALBWACHS, P. CASPI, P. RAYMOND, D. PILAUD. *The Synchronous Data-Flow Programming Language Lustre*, in "Proceedings of the IEEE", vol. 79, n^o 9, September 1991, p. 1305–1320.
- [63] N. HALBWACHS. *Synchronous Programming of Reactive Systems*, Kluwer, 1993.
- [64] 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.
- [65] D. HAREL. *Statecharts: A Visual Formalism for Complex Systems*, in "Science of Computer Programming", vol. 8, 1987, p. 231-274.
- [66] 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.
- [67] B. JEANNET. *Dynamic Partitioning In Linear Relation Analysis. Application To The Verification Of Reactive Systems*, in "Formal Methods in System Design", vol. 23, n^o 1, July 2003, p. 5–37.

-
- [68] 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.
- [69] B. JEANNET, A. LOGINOV, T. REPS, M. SAGIV. *A relational approach to interprocedural shape analysis*, in "Static Analysis Symposium, SAS'04", LNCS, vol. 3148, 2004.
- [70] K. KEUTZER, S. MALIK, A. NEWTON, J. RABAEY, A. SANGIOVANNI-VINCENNELLI. *System Level Design: Orthogonalization of Concerns and Platform-Based Design*, in "IEEE Trans. on Computer-Aided Design", vol. 19, n° 12, 2000.
- [71] 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.
- [72] 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.
- [73] 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.
- [74] 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.
- [75] 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.
- [76] P. J. RAMADGE, W. M. WONHAM. *The Control of Discrete Event Systems*, in "Proceedings of the IEEE", vol. 77, n° 1, 1989.
- [77] 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.